# Expressive Description Logics for Agent-Based Information Retrieval

Ralf Möller, Volker Haarslev and Bernd Neumann
*University of Hamburg, Computer Science Department,*
*Vogt-Kölln-Str. 30, 22527 Hamburg, Germany*

**Abstract.** In this contribution we investigate the use of description logics (DLs) for information retrieval in a multiagent scenario. We first describe two advanced DLs and present the relevant reasoning services provided for information retrieval, in particular instance retrieval, instance checking and example-based instance retrieval. Complete and sound algorithms exist for each of these tasks in both DLs, but it is shown that a combined DL is undecidable. In order to make use of knowledge bases which use different DLs, a broker-based multiagent information retrieval scheme is presented. The main idea is to pose transformed queries to individual agents and combine the answers to obtain a correct but not necessarily complete result. The approach is illustrated with detailed examples.

## 1. Introduction

As more and more information sources of various kinds become available for an increasing number of users, one major challenge for Computer Science is to provide adequate access and retrieval mechanisms. This is not only true for Web-based information which by its nature tends to be highly unorganized and heterogeneous, but also for dedicated databases which are designed to provide a particular service. The guiding example of this paper is a "TV-Assistant" with a database containing TV-program information. The task of the TV-Assistant is to guide TV watchers in selecting "their" favorite program item from a potentially large set of candidates. For example, the TV-Assistant should be able to identify "a pirate movie with sailing ships" among the 300 movies which a new German digital TV channel broadcasts every 30 minutes. Furthermore, based on the preferences of a user, the TV-Assistant also has to determine suitable commercials.

There is obviously a large variety of criteria by which TV watchers would like to express their preferences. They may want to refer to the contents of the program item in terms of its genre type, main characters, location, historical events, plot etc. They may also want to refer to production information, e.g. producer, cast, recording technique, date of origin etc., maybe also to their particular viewer situation, e.g. language and age requirements. While some of these criteria can already be used in existing TV-program services (e.g. genre, cast, age recommendations), content-based retrieval is in its infancy.

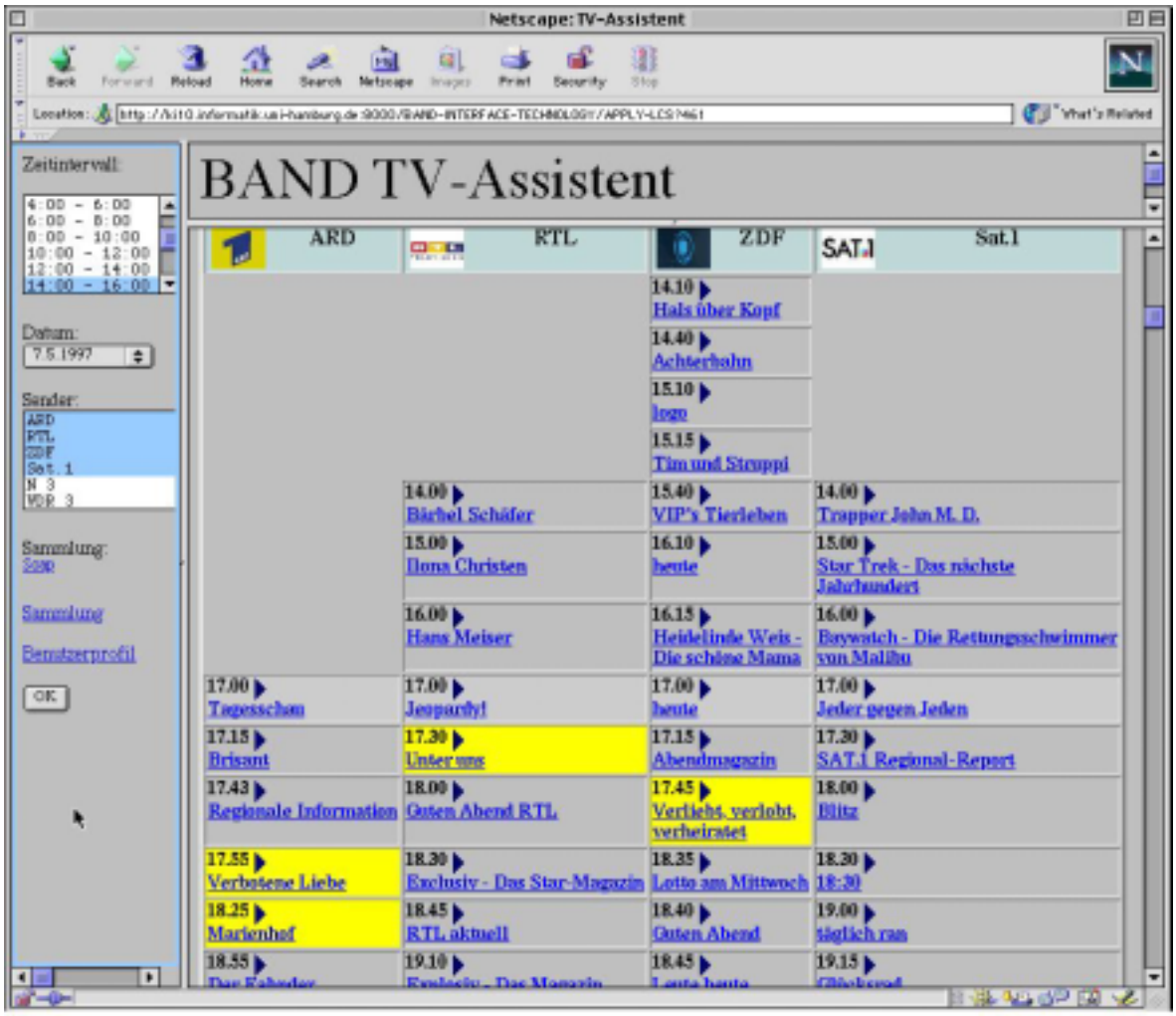



Figure 1: Snapshot of the program table window of the TV-Assistant.

The prevailing approaches for content-based access and retrieval utilize textual information in terms of keywords and word statistics. Surface-based textual information retrieval, typically based on string-indexing, offers several advantages, in particular the use of queries involving natural language terms, and the availability of text documents. On the other hand, string-indexing is unreliable in several respects. First, documents may not be produced with the aim to support textual retrieval. Hence it is a matter of chance whether or not a desirable keyword really appears in the document. Second, as examples of TV-program selections show, naturally expressed queries may involve terms which are less specific than the textual descriptor of the data (or only conceptually close to it), e.g. "sailing ship" instead of "frigate". Similarly, one may be interested in a movie about one's home town, say Hamburg. Content-based retrieval should not only index into descriptors involving the string "Hamburg" but possibly also into locations spatially related to Hamburg, e.g. "Northern Germany" or "Reeperbahn" (Hamburg's famous red-light district). It is also apparent that additional conceptual information must be exploited to avoid unwanted retrieval hits involving certain popular food items.

In this paper we investigate the use of conceptual descriptions based on description logics (DLs) for content-based information retrieval with an agent-based scenario. Our contribution deals with two main aspects. First, strengths and limitations of DLs for information retrieval are investigated. Second, cooperation strategies for agents whose reasoning is based on different DLs are analyzed. When investigating DLs for information retrieval, one can consider a wide variety of languages which have been analyzed in research and partly implemented so far. The most important differentiating aspect of DLs is expressiveness, i.e. what concept expressions may be formulated within a particular DL, and the resulting complexity of inference procedures such as consistency checking or subsumption computations. Hence there are DLs of fairly limited expressiveness but attractive runtime properties, and there are DLs with enriched expressiveness for which one has to pay with doubly exponential inference complexity. Furthermore, if certain restrictions on expressiveness are not observed, a DL may become undecidable in the sense that terminating inference procedures which are complete and sound do not exist any longer. This may be acceptable in some applications, but must be considered a severe disadvantage where reliability is at stake.

An interesting still ongoing development concerns the incorporation of concrete domains into DLs. DLs can be extended to reason about nonsymbolic entities such as real numbers, time intervals, or spatial objects. Again, complexity and decidability considerations impose limitations on the language features. In particular, combining the features of two DLs which are individually decidable may very well result in an undecidable language.

From what we know about DLs today, it seems reasonable to expect that there will not be a single DL optimally suited for all knowledge-based applications. Rather we have to consider heterogeneous special-purpose knowledge bases using DLs of different expressiveness. Each of the knowledge bases may be designed to meet different design

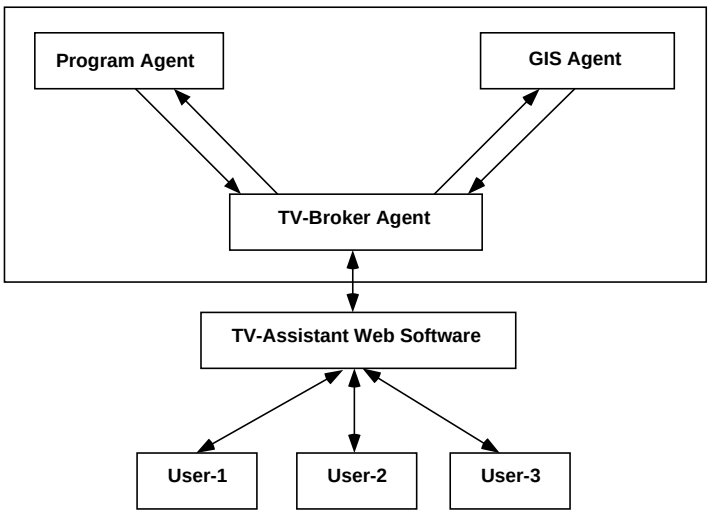



Figure 2: Overview of the System Architecture. A broker agent communicates with an agent which responsible for the tv program and an agent which is a specialist for geographical information system (GIS) requests. The broker agent provides the interface for the software of the TV-Assistant web server.

goals. One knowledge base may provide quick but crude inferences, another may allow more sophisticated inferences at the cost of longer response times, a third one may be a specialist for temporal reasoning, a forth one for spatial reasoning, and so on. This has led us to investigate DLs in an agent-based scenario, where different DLs are organized to cooperate in an information retrieval task.

The general structure of an agent-based scenario is shown in Figure 2. With a web browser, different users interact with a TV program information system (see Figure 1 for an interface). The web server submits information retrieval tasks to an information broker which is the central node in the agent network. The broker may invoke services of other agent nodes which are his subagents. The knowledge bases of the agents may be based on different DLs and may have been developed independently from each other. In order to communicate with his subagents, the broker must have certain interschema knowledge to formulate an information retrieval task for a subagent and make use of his result. Note that the user of the TV program information system is not concerned with the broker's activities and it's associated inferences. Users interact with the system via interfaces such as those depicted in Figure 1.

In the context of DL knowledge bases we will consider instance retrieval and instance checking as basic information retrieval tasks. Instance retrieval is a well-known inference service of a DL system where a concept term is submitted as a query, and the task is to retrieve all individuals which are instances of the concept term. For instance checking, a query comprises a concept term together with a set of individuals which are to be checked against the concept term. Both tasks are identical from a logical perspective and amount to consistency checking. We will also consider an example-based

extension of instance retrieval, where examples are used to compute a second concept term of which the retrieved individuals have to be instances.

In DLs, individuals are represented as assertional descriptions in so-called ABoxes. An individual description relates an individual identifier to the concepts and the roles of the terminology of a domain. The terminology is defined in the so-called TBox. Since all reasoning processes of an agent are based on the agent's terminology, it is obvious that instance checking can only be performed by an agent if the task is formulated in terms of the agent's own terminology. In our scenario, it is the task of the broker to transform queries or subtasks thereof so that subagents can be employed. For this purpose, the broker is equipped with interschema knowledge which relates different terminologies to each other. Thus, the broker can approximate assertional descriptions and concept terms by expressions which only use the concepts and roles of a particular subagent. In exceptional cases, these expressions may be equivalent, but in general they will be approximations. Hence it is an interesting question whether subagent reasoning can be employed at all for instance checking without jeopardizing the correctness of the overall result.

In our contribution we show that this is indeed possible - albeit only to some extent. The basic idea is to approximate queries in such a way that the concept term of a query (which constrains individuals) is specialized, and individual descriptions - if there are any - are generalized. Thus, if approximated individuals are found to be instances of the approximated concept term, the original individuals will also be instances of the original concept term. On the other hand, we have to be aware of the fact that the combined logics of two agents may be undecidable. If this is so, inconsistency may not be detected reliably by any procedure which combines the knowledge of these agents. This is an inherent limitation with interesting consequences regarding the inferences which one can compute in a multi-agent information retrieval scenario.

The remainder of this contribution is organized as follows. In Section 2 we present the formal foundations of two advanced DLs and of the inference processes supported by these languages. For one of the DLs, $\mathcal{ALCNH}_{R^+}$,[1] a highly optimized ABox reasoner has been implemented providing a powerful infrastructure for realistic applications. The other DL, $\mathcal{ALCRP}(\mathcal{RCC})$, has been designed to provide spatioterminological reasoning. This competence cannot directly be added to $\mathcal{ALCNH}_{R^+}$ because the resulting language would be undecidable. In Section 2 we present several introductory examples to illustrate the type of reasoning which is supported by each of the DLs.

In Section 3 we develop the framework for multiagent information retrieval. In particular, we present multiagent inference services and the transformations of queries required for cooperative information retrieval. Although some limitations are unavoidable, a sound algorithm can be presented for example-based instance retrieval and instance checking queries. The algorithm exploits the knowledge and reasoning power of specialists.

In Section 4 we present a detailed example involving a broker agent, a program

[1]The pronunciation of $\mathcal{ALCNH}_{R^+}$ is ALC-nature.

agent and an agent with access to a geographical information system (GIS). The latter agent is called GIS Agent. First, a user submits an example-based instance retrieval query to the broker and the broker invokes the program agent to provide the answer. Simultaneously, the retrieved program items are submitted for instance checking in order to trigger certain advertisements. The instance checking task involves spatial reasoning and is delegated to the GIS agent who is a specialist for spatial reasoning. Altogether, three different DL agents are shown to cooperatively solve a realistic information retrieval task.

## 2. Description Logics: A Short Introduction

In the field of formal inference systems, description logics have been proven to be a sound basis for solving application problems. Detailed introductions to description logics can be found in [35, 12, 4]. The relation of description logics to logic programming techniques is described in [1].

In description logics, the main notions for domain modeling are concepts (unary predicates) and roles. In most DLs roles are binary predicates (but see [7] for a DL with n-ary roles). A set of axioms (TBox) is used for modeling the terminology of an application. Knowledge about specific individuals and their interrelationships is modeled with a set of additional axioms (so-called ABox).

The description logic $\mathcal{ALC}$ [33] is a propositionally complete representation language providing conjunction, universal quantification and full negation. As an introductory example, let us consider family relationships. For instance, a woman is a human whose gender is restricted to be female whereas a man is a human whose gender is restricted to be male. The concepts male and female are disjoint. A parent is a human which has at least one child which must be a human. Furthermore, a mother is a woman and a parent. This kind of terminological knowledge can easily be modeled with concepts and roles in the language $\mathcal{ALC}$.

$$\begin{aligned}
\textbf{woman} &\doteq \mathsf{human} \sqcap \forall\, \mathsf{has_gender} \,.\, \mathsf{female} \\
\textbf{man} &\doteq \mathsf{human} \sqcap \forall\, \mathsf{has_gender} \,.\, \mathsf{male} \\
\textbf{parent} &\doteq \mathsf{human} \sqcap \exists\, \mathsf{has_child} \,.\, \mathsf{human} \\
\textbf{mother} &\doteq \mathsf{woman} \sqcap \mathsf{parent} \\
\textbf{male} &\sqsubseteq \neg \mathsf{female}
\end{aligned}$$

The first four declarations declare necessary and sufficient conditions for the concepts on the left-hand side. In the last declaration only necessary conditions are provided for male which is declared to be disjoint to female. From this it follows that the concepts man and woman are disjoint as well.

The description logic $\mathcal{ALCN}\mathcal{H}_{R^+}$ extends $\mathcal{ALC}$ with number restrictions, role hierarchies, and transitively closed roles. For instance, in the family example, we assume a role has_descendant which is declared to be transitive. A direct descendant can be

distinguished using role hierarchies, i.e. a non-transitive role has_child is declared with has_descendant as a "superrole".

$$\textbf{has\_child} \sqsubseteq \text{has\_descendant}$$

Concept terms can be composed to describe complex conceptual notions. A "queen with a small family" can be defined as a mother who has at most two children and whose descendants are either princes or princesses. This type of queen can be described with the term:

$$\begin{aligned}
&\textbf{queen_with_small_family} \sqsubseteq \\
&\quad \text{mother} \sqcap \\
&\quad \exists_{\leq 2}\, \text{has_child} \sqcap \\
&\quad \forall\, \text{has_descendant} \,.\, (\text{prince} \sqcup \text{princess}) \\
&\textbf{prince} \sqsubseteq \text{man} \\
&\textbf{princess} \sqsubseteq \text{woman}
\end{aligned}$$

In this case only necessary conditions for the concepts queen_with_small_family, prince and princess are provided.

Now, let us turn to reasoning about individuals. If there exists an individual i1 which has a child i2 which, in turn has a child i3, then, due to has_descendent being transitive and a "superrole" of has_child, i3 is implicitly declared a descendant of i1.

$$\begin{aligned}
&(\text{i1}, \text{i2}) : \text{has_child} \\
&(\text{i2}, \text{i3}) : \text{has_child} \\
&\quad \text{i1} : \text{queen_with_small_family} \\
&\quad \text{i1} : \forall\, \text{has_descendant} \,.\, \forall\, \text{has_gender} \,.\, \text{female}
\end{aligned}$$

Moreover, since the individual i1 is known to be an instance of the above-mentioned concept queen_with_small_family, the individual i3 can be inferred to be an instance of (prince $\sqcup$ princess). With the assertional axiom $\text{i1} : \forall\, \text{has\_descendant} \,.\, \forall\, \text{has\_gender} \,.\, \text{female}$ it can be proven that i3 is an instance of princess because due to the disjointness of male and female the alternative prince of the disjunction prince $\sqcup$ princess is ruled out.

## 3. The Description Logics $\mathcal{ALCNH}_{R^+}$ and $\mathcal{ALCRP}(\mathcal{D})$

In this section we present the formal foundations of two advanced DLs and of the inference processes supported by these languages. This section also includes further examples for knowledge modeling in description logics.

### 3.1. The Description Logic $\mathcal{ALCNH}_{R^+}$

We begin with a presentation of the DL $\mathcal{ALCNH}_{R^+}$ which has been developed in the research group of the authors. $\mathcal{ALCNH}_{R^+}$ is an expressive description logic for which optimization techniques are known that allow for the implementation of a practical knowledge representation system [17].

#### 3.1.1. The Concept Language of $\mathcal{ALCNH}_{R^+}$

We present the syntax and semantics of the language $\mathcal{ALCNH}_{R^+}$ for specifying concept and role inclusions.

**Definition 1 (Role Inclusions)** Let $P$ and $T$ be disjoint sets of non-transitive and transitive *role* names, respectively, and let $R$ be defined as $R = P \cup T$. Let $\mathsf{R}$ and $\mathsf{S}$ be role names, then $\mathsf{R} \sqsubseteq \mathsf{S}$ *(role inclusion axiom)* is a terminological axiom. The role $\mathsf{R}$ is called a subrole of $\mathsf{S}$ and $\mathsf{S}$ is a superrole of $\mathsf{R}$.

Additionally we define the set of ancestors and descendants of a role.

**Definition 2 (Role Descendants/Ancestors)** Let $\sqsubseteq^*$ be the transitive reflexive closure of $\sqsubseteq$. Given a role hierarchy $\sqsubseteq^*$ the set $\mathsf{R}^{\uparrow} := \{\mathsf{S} \in R \mid \mathsf{R} \sqsubseteq^* \mathsf{S}\}$ defines the *ancestors* and $\mathsf{R}^{\downarrow} := \{\mathsf{S} \in R \mid \mathsf{S} \sqsubseteq^* \mathsf{R}\}$ the *descendants* of a role $\mathsf{R}$. We also define the set $S := \{\mathsf{R} \in P \mid \mathsf{R}^{\downarrow} \cap T = \emptyset\}$ of *simple* roles that are neither transitive nor have a transitive role as descendant.

**Definition 3 (Concept Terms)** Let $C$ be a set of *concept names* (also called atomic concepts) which is disjoint from $R$. Any element of $C$ is a *concept term*. If $\mathsf{C}$ and $\mathsf{D}$ are concept terms, $\mathsf{R} \in R$ is an arbitrary role, $\mathsf{S} \in S$ is a simple role, $n > 1$, and $m > 0$, then the following expressions are also concept terms:

- $\mathsf{C} \sqcap \mathsf{D}$ *(conjunction)*
- $\mathsf{C} \sqcup \mathsf{D}$ *(disjunction)*
- $\neg\mathsf{C}$ *(negation)*
- $\forall\, \mathsf{R}\,.\,\mathsf{C}$ *(concept value restriction)*
- $\exists\, \mathsf{R}\,.\,\mathsf{C}$ *(concept exists restriction)*
- $\exists_{\leq m}\, \mathsf{S}$ *(at most number restriction)*
- $\exists_{\geq n}\, \mathsf{S}$ *(at least number restriction).*

The terms $\top$ ($\bot$) are an abbreviation for $\mathsf{C} \sqcup \neg\mathsf{C}$ ($\mathsf{C} \sqcap \neg\mathsf{C}$). For an arbitrary role $\mathsf{R}$, the term $\exists_{\geq 1}\, \mathsf{R}$ can be rewritten as $\exists\, \mathsf{R}\,.\,\top$, $\exists_{\geq 0}\, \mathsf{R}$ as $\top$, and $\exists_{\leq 0}\, \mathsf{R}$ as $\forall\, \mathsf{R}\,.\,\bot$. Thus, we do not consider these terms as number restrictions in our language.

The concept language syntactically restricts the combination of number restrictions and transitive roles. Number restrictions are only allowed for *simple* roles.

**Definition 4 (Generalized Concept Inclusions, TBox)** If $\mathsf{C}$ and $\mathsf{D}$ are concept terms, then $\mathsf{C} \sqsubseteq \mathsf{D}$ *(generalized concept inclusion* or *GCI)* is a terminological axiom as

well. $\mathsf{C} \doteq \mathsf{D}$ is abbreviation for the two GCIs $\mathsf{C} \sqsubseteq \mathsf{D}$, $\mathsf{D} \sqsubseteq \mathsf{C}$. A finite set of terminological axioms $\mathcal{T}$ is called a *terminology* or *TBox*.

The next definition provides a model-theoretic semantics for the language introduced above.

**Definition 5 (Semantics)** An *interpretation* $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$ consists of a set $\Delta^{\mathcal{I}}$ (the domain) and an interpretation function $\cdot^{\mathcal{I}}$. The interpretation function maps each concept name $\mathsf{C}$ to a subset $\mathsf{C}^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, each role name $\mathsf{R}$ to a subset $\mathsf{R}^{\mathcal{I}}$ of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$. Let the symbols $\mathsf{C}$, $\mathsf{D}$ be concept expressions, and $\mathsf{R}$, $\mathsf{S}$ be role names. Then the interpretation function can be extended to arbitrary concept and role terms as follows ($\|\cdot\|$ denotes the cardinality of a set):

$$
\begin{aligned}
(\mathsf{C} \sqcap \mathsf{D})^{\mathcal{I}} &:= \mathsf{C}^{\mathcal{I}} \cap \mathsf{D}^{\mathcal{I}} \\
(\mathsf{C} \sqcup \mathsf{D})^{\mathcal{I}} &:= \mathsf{C}^{\mathcal{I}} \cup \mathsf{D}^{\mathcal{I}} \\
(\neg \mathsf{C})^{\mathcal{I}} &:= \Delta^{\mathcal{I}} \setminus \mathsf{C}^{\mathcal{I}} \\
(\exists\, \mathsf{R}\,.\,\mathsf{C})^{\mathcal{I}} &:= \{a \in \Delta^{\mathcal{I}} \mid \exists\, b \in \Delta^{\mathcal{I}} : (a,b) \in \mathsf{R}^{\mathcal{I}}, b \in \mathsf{C}^{\mathcal{I}}\} \\
(\forall\, \mathsf{R}\,.\,\mathsf{C})^{\mathcal{I}} &:= \{a \in \Delta^{\mathcal{I}} \mid \forall\, b \in \Delta^{\mathcal{I}} : (a,b) \in \mathsf{R}^{\mathcal{I}} \Rightarrow b \in \mathsf{C}^{\mathcal{I}}\} \\
(\exists_{\geq n}\, \mathsf{R})^{\mathcal{I}} &:= \{a \in \Delta^{\mathcal{I}} \mid \|\{b \mid (a,b) \in \mathsf{R}^{\mathcal{I}}\}\| \geq n\} \\
(\exists_{\leq n}\, \mathsf{R})^{\mathcal{I}} &:= \{a \in \Delta^{\mathcal{I}} \mid \|\{b \mid (a,b) \in \mathsf{R}^{\mathcal{I}}\}\| \leq n\}
\end{aligned}
$$

An interpretation $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$ iff it satisfies (1) $\mathsf{C}^{\mathcal{I}} \subseteq \mathsf{D}^{\mathcal{I}}$ for all terminological axioms (GCIs) $\mathsf{C} \sqsubseteq \mathsf{D}$ in $\mathcal{T}$ and $\mathsf{R}^{\mathcal{I}} \subseteq \mathsf{S}^{\mathcal{I}}$ for all terminological axioms $\mathsf{R} \sqsubseteq \mathsf{S}$ (role inclusions) in $\mathcal{T}$, and (2) iff for every $\mathsf{R} \in T : \mathsf{R}^{\mathcal{I}} = (\mathsf{R}^{\mathcal{I}})^{+}$.

Description logics can be used for modeling the knowledge required for an application domain. Once a knowledge base has been defined, queries of different types can be answered. The different query types are defined as inference problems.

**Definition 6 (Concept and TBox Inference Problems)** A concept term $\mathsf{C}$ is *consistent* w.r.t. a TBox $\mathcal{T}$ iff there exists a model $\mathcal{I}$ of $\mathcal{T}$ such that $\mathsf{C}^{\mathcal{I}} \neq \emptyset$. A concept term $\mathsf{C}$ *subsumes* a concept term $\mathsf{D}$ w.r.t. a TBox $\mathcal{T}$ (written $\mathsf{D} \preceq_{\mathcal{T}} \mathsf{C}$), iff $\mathsf{D}^{\mathcal{I}} \subseteq \mathsf{C}^{\mathcal{I}}$ for all models $\mathcal{I}$ of $\mathcal{T}$. The problem of computing the most-specific atomic concepts mentioned in a TBox $\mathcal{T}$ which subsume a certain concept is known as computing the *parents* of a concept (w.r.t. a TBox $\mathcal{T}$). The *children* are the most-general atomic concepts (mentioned in a TBox $\mathcal{T}$) which are subsumed by a certain concept.[2] The *concept descendants* (*concept ancestors*) relation between concepts is defined to be the transitive closure of the *children* (*parents*) relation. The set of *fillers* of a role $\mathsf{R}$ w.r.t. an element $\mathsf{x}$ of the domain $\Delta^{\mathcal{I}}$ is defined as $\{\mathsf{y} \in \Delta^{\mathcal{I}} \mid (\mathsf{x},\mathsf{y}) \in \mathsf{R}^{\mathcal{I}}\}$.

[2]Note that the parents and children might become more specific if terminological axioms are added to a TBox.

In $\mathcal{ALCNH}_{R^+}$, satisfiability and subsumption can be mutually reduced to each other since $\mathsf{C} \preceq_{\mathcal{T}} \mathsf{D}$ iff $\mathsf{C} \sqcap \neg \mathsf{D}$ is not satisfiable w.r.t. $\mathcal{T}$ and $\mathsf{C}$ is unsatisfiable w.r.t. $\mathcal{T}$ iff $\mathsf{C} \preceq_{\mathcal{T}} \bot$.

The computation of the parents and children of every atomic concepts in a TBox is also called classification of a TBox. This inference service is needed to build a hierarchy of concept names w.r.t. specificity. The concept hierarchy is usually used to verify the contents of a TBox during the development phase.

### 3.1.2. An Example for TBox-Reasoning with $\mathcal{ALCNH}_{R^+}$

We begin our example by specifying some terminological axioms.

$$\textbf{captain} \sqsubseteq \mathsf{person}$$

$$\textbf{ship} \sqsubseteq \forall\, \mathsf{has\_home\_port} \,.\, \mathsf{port} \sqcap \forall\, \mathsf{has\_captain} \,.\, \mathsf{captain}$$

$$\textbf{ship\_with\_captain} \doteq \mathsf{ship} \sqcap \exists_{\geq 1}\, \mathsf{has\_captain} \sqcap \exists_{\leq 1}\, \mathsf{has\_captain}$$

$$\textbf{ship\_with\_cargo} \doteq$$
$$\mathsf{ship} \sqcap \exists_{\geq 1}\, \mathsf{has\_captain} \sqcap \exists_{\leq 1}\, \mathsf{has\_captain} \sqcap$$
$$\exists_{\geq 1}\, \mathsf{has\_cargo\_storage}$$

$$\textbf{shipyard} \sqsubseteq \forall\, \mathsf{has\_ship\_in\_repair\_dock} \,.\, \mathsf{ship\_in\_shipyard}$$

The first terminological axiom declares a subsumption relationship between atomic concepts. All captains are persons. The axiom for ship specifies that each filler of the attribute has_home_port must be a port, i.e. in DL terminology, each filler must be an instance of the concept port. Furthermore, each filler of the attribute has_captain must be a captain. Though not explicitly stated it is evident that, due to the semantics given above, ship_with_captain subsumes ship_with_cargo. That is to say, ship_with_cargo is a subconcept of ship_with_captain. Not all subsumption relationships are that obvious, as the following example indicates.

$$\textbf{has\_container\_storage} \sqsubseteq \mathsf{has\_cargo\_storage}$$

$$\textbf{has\_cooling\_storage} \sqsubseteq \mathsf{has\_container\_storage}$$

$$\textbf{has\_gas\_storage} \sqsubseteq \mathsf{has\_container\_storage}$$

The role has_container_storage is a subrole of has_cargo_storage. Afterwards, two subroles for has_container_storage are defined: has_cooling_storage and has_gas_storage. Then, the following concept axioms are added to the TBox.

$$\textbf{container\_ship} \sqsubseteq \mathsf{ship}$$

$$\exists\, \mathsf{has\_container\_storage} \,.\, \top \sqsubseteq \mathsf{container\_ship}$$

$$\top \sqsubseteq \forall\, \mathsf{has\_container\_storage} \,.\, \mathsf{container}$$

$$\top \sqsubseteq \forall\, \mathsf{has\_cooling\_storage} \,.\, \mathsf{cooled\_container}$$

$$\top \sqsubseteq \forall\, \mathsf{has\_gas\_storage} \,.\, \mathsf{gas\_container}$$

$$\begin{aligned}&\textsf{type\_1\_ship} \sqsubseteq \\ &\quad \exists_{\leq 1000}\, \textsf{has\_container\_storage} \sqcap \\ &\quad \exists_{\geq 600}\, \textsf{has\_cooling\_storage} \sqcap \\ &\quad \exists_{\geq 600}\, \textsf{has\_gas\_storage} \sqcap\end{aligned}$$

A container_ship is a ship. Moreover, the second axiom declares so-called domain restriction for the role has_container_storage. If there are any individuals set into relation to an individual i via the role has_container_storage (i.e. i is on the left-hand side), then the individual i must be a container_ship. Afterwards, three range restrictions are defined. For any individual it holds that the fillers of the role has_container_storage must be instances of container, the fillers of has_cooling_storage must be instances of the concept cooled_container, and the fillers of has_gas_storage must be instances of gas_container. The last axiom specifies a type_1_ship as an object with up to 1000 fillers for has_container_storage, at least 600 fillers for has_cooling_storage and at least 600 fillers for has_gas_storage.

Now let us assume that due to safety problems, container ships with storage capacity for containers which can be cooled and which may contain gas are classified as a dangerous ships. For instance, the following axiom is added to the TBox in order to define a concept dangerous_ship with necessary and sufficient conditions.

$$\begin{aligned}&\textsf{dangerous\_ship} \doteq \\ &\quad (\exists\, \textsf{has\_container\_storage}\,.\,(\textsf{cooled\_container} \sqcap \textsf{gas\_container})) \sqcup \\ &\quad (\exists\, \textsf{has\_cargo\_storage}\,.\,\textsf{toxic\_waste})\end{aligned}$$

After TBox classification, non-obvious subsumption relationships become apparent. Due to the domain restriction for has_container_storage, the concept dangerous_ship is inferred to be an instance of container_ship. Furthermore, and more interesting, it can be deduced that the concept type_1_ship is subsumed by dangerous_ship. Since there can be at most 1000 fillers for has_container_storage, there must exist at least 200 individuals which are fillers of both roles has_cooling_storage and gas_container. Hence, there must be at least 200 fillers of has_container_storage which are instances of the concept cooled_container $\sqcap$ gas_container. Classifying the TBox and asking for the concept descendants of dangerous_ship yields type_1_ship, possibly among others.

A container itself could include dangerous things. For instance, a certain class of containers with toxic waste (but with an innocent name) is defined.

$$\begin{aligned}&\textsf{type\_47\_container} \sqsubseteq \textsf{container} \sqcap \exists\, \textsf{has\_cargo\_storage}\,.\,\textsf{toxic\_waste} \\ &\textsf{type\_2\_ship} \sqsubseteq \exists\, \textsf{has\_container\_storage}\,.\,\textsf{type\_47\_container}\end{aligned}$$

Unfortunately, with the axioms given above type_2_ship would not be classified as a dangerous_ship. This would be the case, however, if has_cargo_storage were declared as a transitive role. As indicate above, this facility is also supported by $\mathcal{ALCN}\mathcal{H}_{R^+}$.

### 3.1.3. The Assertional Language of $\mathcal{ALCNH}_{R^+}$

In the following, the language for representing knowledge about individuals is introduced.

**Definition 7 (Assertional Axioms, ABox)** An *ABox* $\mathcal{A}$ is a finite set of assertional axioms which are defined as follows: Let $O$ be a set of individual names. If $\mathsf{C}$ is a concept term, $\mathsf{R}$ a role name, and $\mathsf{a}, \mathsf{b} \in O$ are individual names, then the following expressions are *assertional axioms*:

- $\mathsf{a} : \mathsf{C}$ *(concept assertion)*,
- $(\mathsf{a}, \mathsf{b}) : \mathsf{R}$ *(role assertion)*.

The interpretation function $\cdot^{\mathcal{I}}$ of the interpretation $\mathcal{I}$ for the concept language can be extended to the assertional language by additionally mapping every individual name from $O$ to a single element of $\Delta^{\mathcal{I}}$ in a way such that for $\mathsf{a}, \mathsf{b} \in O$, $\mathsf{a}^{\mathcal{I}} \neq \mathsf{b}^{\mathcal{I}}$ if $\mathsf{a} \neq \mathsf{b}$ *(unique name assumption)*. This ensures that different individuals in $O$ are interpreted as different objects. An interpretation satisfies an assertional axiom $\mathsf{a} : \mathsf{C}$ iff $\mathsf{a}^{\mathcal{I}} \in \mathsf{C}^{\mathcal{I}}$ and $(\mathsf{a}, \mathsf{b}) : \mathsf{R}$ iff $(\mathsf{a}^{\mathcal{I}}, \mathsf{b}^{\mathcal{I}}) \in \mathsf{R}^{\mathcal{I}}$.

An interpretation is a *model* of an ABox $\mathcal{A}$ w.r.t. a TBox $\mathcal{T}$ iff it is a model of $\mathcal{T}$ and furthermore satisfies all assertional axioms in $\mathcal{A}$.

**Definition 8 (ABox Inference Problems)** The *ABox consistency problem* is to decide whether a given ABox $\mathcal{A}$ is consistent w.r.t. a TBox $\mathcal{T}$. An ABox is *consistent* w.r.t. a TBox $\mathcal{T}$ iff it has a model w.r.t. $\mathcal{T}$. Otherwise the ABox is called *inconsistent*.

The test whether an individual $\mathsf{a}$ is an instance of a concept term $\mathsf{C}$ w.r.t. an ABox $\mathcal{A}$ and a TBox $\mathcal{T}$ is called *instance checking*.

The most-specific atomic concepts of which an individual is an instance are called the *direct types* of the individual (w.r.t. an ABox $\mathcal{A}$).[3] This problem can be reduced to subsequent instance problems.

The set of *fillers* of a role $\mathsf{R}$ w.r.t. an individual $\mathsf{i}$ in an ABox $\mathcal{A}$ is defined as $\{\mathsf{x} \mid \mathcal{A} \models (\mathsf{i}, \mathsf{x}) : \mathsf{R}\}$.

Consistency of concept terms can be reduced to ABox consistency as follows: A concept term $\mathsf{C}$ is consistent iff the ABox $\{\mathsf{a} : \mathsf{C}\}$ is consistent. For description logics which support negation for concept terms, this *instance problem* can be reduced to the problem of deciding if the ABox $\mathcal{A} \cup \{\mathsf{a} : \neg\mathsf{C}\}$ is inconsistent. Let $\mathsf{Test}$ be an atomic concept not mentioned in $\mathcal{A}$ and its associated TBox $\mathcal{T}$. It is easy to verify that determining the filler of a role w.r.t. an individual can be reduced to checking for all individuals $\mathsf{x}$ mentioned in an ABox $\mathcal{A}$ whether the ABox $\mathcal{A}_{\mathcal{T}} \cup \{\mathsf{x} : \neg\mathsf{Test}, \mathsf{i} : \forall\, \mathsf{R}\,.\,\mathsf{Test}\}$ is inconsistent.

For the sake of completeness we also give a formal definition of a knowledge base.

**Definition 9 (Knowledge Base)** A knowledge base is a tuple $(\mathcal{T}, \mathcal{A})$ where $\mathcal{T}$ is a TBox and $\mathcal{A}$ is an ABox.

[3] Again, th the direct types might change if assertional axioms are added to the ABox.

### 3.1.4. An Example for ABox-Reasoning with $\mathcal{ALCNH}_{R^+}$

Now let us continue the example from above and introduce some individuals in an ABox. In some circumstances an individual should be an instance of a concept when the instance is set into relation to another instance. For instance, a ship individual should be an instance of the concept ship_in_shipyard when it is set into relation to a shipyard. A similar situation occurs when persons become customers if they are set into relation to a bank. This dynamic classification is no problem when ABoxes are used for object representation. The ship example is continued with the following assertions.

$$\mathsf{s1} : \mathsf{ship},\ \mathsf{yard1} : \mathsf{shipyard},\ (\mathsf{yard1}, \mathsf{s1}) : \mathsf{has\_ship\_in\_repair\_dock}$$

Even though s1 is "created" as a ship, asking for the direct types of s1 reveals that s1 is also an instance of ship_in_shipyard because s1 is set into relation to yard1 via has_ship_in_repair_dock and, due to the axiom for shipyard in the TBox, for each shipyard *all* fillers of has_ship_in_repair_dock are instances of ship_in_shipyard. This instance classification is "dynamic" because s1 will no longer be a ship_in_shipyard when the related statement is retracted.

After asserting the ABox statements $(\mathsf{s1}, \mathsf{c1}) : \mathsf{has\_captain}$ and $\mathsf{s1} : \exists_{\leq 1}\ \mathsf{has\_captain}$, the ship s1 is automatically classified as a ship_with_captain because the sufficient conditions for ship_with_captain are fulfilled (see also the axiom for ship presented above).

## 3.2. The Description Logic $\mathcal{ALCRP}(\mathcal{D})$

We now present the syntax and semantics of the language $\mathcal{ALCRP}(\mathcal{D})$. This is a language with less expressive power than $\mathcal{ALCNH}_{R^+}$ regarding some constructs such as number restrictions, but more powerful because of the incorporation of so-called concrete domains (see below).

### 3.2.1. The Concept Language of $\mathcal{ALCRP}(\mathcal{D})$

We present the syntax and semantics of the language for specifying concept and role inclusions. In accordance with [2] we also define the notion of a concrete domain.

**Definition 10 (Concrete Domain)** A *concrete domain* $\mathcal{D}$ is a pair $(\Delta_\mathcal{D}, \Phi_\mathcal{D})$, where $\Delta_\mathcal{D}$ is a set called the domain, and $\Phi_\mathcal{D}$ is a set of predicate names. Each predicate name $\mathsf{P}_\mathcal{D}$ from $\Phi_\mathcal{D}$ is associated with an arity $n$ and an $n$-ary predicate $\mathsf{P}_\mathcal{D} \subseteq \Delta_\mathcal{D}^n$. A concrete domain $\mathcal{D}$ is called *admissible* iff:

- The set of predicate names $\Phi_\mathcal{D}$ is closed under negation and $\Phi_\mathcal{D}$ contains a name $\top_\mathcal{D}$ for $\Delta_\mathcal{D}$,
- The satisfiability problem $\mathsf{P}_1^{\mathsf{n}_1}(\mathsf{x}_{11}, \ldots, \mathsf{x}_{1\mathsf{n}_1}) \wedge \ldots \wedge \mathsf{P}_\mathsf{m}^{\mathsf{n}_\mathsf{m}}(\mathsf{x}_{\mathsf{m}1}, \ldots, \mathsf{x}_{\mathsf{m}\mathsf{n}_\mathsf{m}})$ is decidable ($\mathsf{m}$ is finite, $\mathsf{P}_\mathsf{i}^{\mathsf{n}_\mathsf{i}} \in \Phi_\mathcal{D}$, and $\mathsf{x}_{\mathsf{j}\mathsf{k}}$ is a name for an object from $\Delta_\mathcal{D}$).

**Definition 11 (Role Terms)** Let $R$ and $F$ be disjoint sets of role and feature names, respectively. Any element of $R \cup F$ is an *atomic* role term. A composition of features (written $\mathsf{f_1 f_2 \ldots f_n}$) is called a feature chain. A simple feature can be considered as a feature chain of length 1. If $\mathsf{P} \in \Phi_{\mathcal{D}}$ is a predicate name with arity $n + m$ and $\mathsf{u_1}, \mathsf{u_2}, \ldots, \mathsf{u_n}$ as well as $\mathsf{v_1}, \mathsf{v_2}, \ldots, \mathsf{v_m}$ are feature chains, then the expression $\exists(\mathsf{u_1}, \ldots, \mathsf{u_n})(\mathsf{v_1}, \ldots, \mathsf{v_m}).\mathsf{P}$ (*role-forming predicate operator*) is a *complex* role term. Let $\mathsf{S}$ be a role name and let $\mathsf{T}$ be a role term. Then $\mathsf{S} \doteq \mathsf{R}$ is a *terminological axiom*. This type of terminological axiom is also called *role introduction*.

Using the definitions from above, we define the syntax of concept terms in $\mathcal{ALCRP}(\mathcal{D})$.

**Definition 12 (Concept Terms)** Let $C$ be a set of concept names which is disjoint from $R$ and $F$. Any element of $C$ is a *concept term*. If $\mathsf{C}$ and $\mathsf{D}$ are concept terms, $\mathsf{R} \in R$ is an arbitrary role, $\mathsf{S} \in S$ is a simple role, $\mathsf{P} \in \Delta_{\mathcal{D}}$ is a predicate of the concrete domain, $\mathsf{u_i}$ is a feature chain, $n > 1$, and $m > 0$, then the following expressions are also concept terms:

- $\mathsf{C} \sqcap \mathsf{D}$ *(conjunction)*
- $\mathsf{C} \sqcup \mathsf{D}$ *(disjunction)*
- $\neg\mathsf{C}$ *(negation)*
- $\forall\, \mathsf{R}\,.\,\mathsf{C}$ *(concept value restriction)*
- $\exists\, \mathsf{R}\,.\,\mathsf{C}$ *(concept exists restriction)*
- $\exists\, \mathsf{u_1}, \ldots, \mathsf{u_n}\,.\,\mathsf{P}$ *(predicate exists restriction).*

A concept term may be put in parentheses. $\top$ ($\bot$) is considered as an abbreviation for $\mathsf{C} \sqcup \neg\mathsf{C}$ ($\mathsf{C} \sqcap \neg\mathsf{C}$).

**Definition 13 (Concept Introduction Axioms, TBox)** Let $\mathsf{A}$ be a concept name and let $\mathsf{D}$ be a concept term. Then $\mathsf{A} \doteq \mathsf{D}$ and $\mathsf{A} \sqsubseteq \mathsf{D}$ are terminological axioms as well. A finite set of terminological axioms $\mathcal{T}$ is called a *terminology* or *TBox* if the left-hand sides of all terminological axioms in $\mathcal{T}$ are unique and, furthermore, all concept definitions are acyclic. The axioms $\mathsf{A} \sqsubseteq \mathsf{D}$ in a TBox are also called *concept introduction axioms*.

The next definition provides a model-theoretic semantics for the language introduced above. Let $\mathcal{D} = (\Delta_{\mathcal{D}}, \Phi_{\mathcal{D}})$ be a concrete domain.

**Definition 14 (Semantics)** An *interpretation* $\mathcal{I}_{\mathcal{D}} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})_{\mathcal{D}}$ consists of a set $\Delta^{\mathcal{I}}$ (the abstract domain), a set $\Delta^{\mathcal{D}}$ (the domain of the 'concrete domain' $\mathcal{D}$) and an interpretation function $\cdot^{\mathcal{I}}$. The interpretation function $\cdot^{\mathcal{I}}$maps each concept name $\mathsf{C}$ to a subset $\mathsf{C}^{\mathcal{I}}$ of $\Delta^{\mathcal{I}}$, each role name $\mathsf{R}$ from $R$ to a subset $\mathsf{R}^{\mathcal{I}}$ of $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, each feature $\mathsf{f}$ from $F$ to a partial function $\mathsf{f}^{\mathcal{I}}$ from $\Delta^{\mathcal{I}}$ to $\Delta^{\mathcal{I}} \cup \Delta^{\mathcal{D}}$, and each predicate name $\mathsf{P}$ from $\Phi_{\mathcal{D}}$ with arity $n$ to a subset $\mathsf{P}^{\mathcal{I}}$ of $\Delta^n_{\mathcal{D}}$. If $\mathsf{u} = \mathsf{f_1} \cdots \mathsf{f_n}$ is a feature chain, then $\mathsf{u}^{\mathcal{I}}$ denotes the composition $\mathsf{f_1}^{\mathcal{I}} \circ \ldots \circ \mathsf{f_n}^{\mathcal{I}}$ of partial functions $\mathsf{f_1}^{\mathcal{I}}$, ... ,$\mathsf{f_n}^{\mathcal{I}}$. Let the symbols $\mathsf{C}$, $\mathsf{D}$ be concept expressions, $\mathsf{R}$, $\mathsf{S}$ be role names, $\mathsf{u_1}, \ldots, \mathsf{u_n}$ be feature chains and let $\mathsf{P}$

be a predicate name. Then, the interpretation function can be extended to arbitrary concept and role terms as follows.

$$
\begin{aligned}
(\mathsf{C} \sqcap \mathsf{D})^{\mathcal{I}} &:= \mathsf{C}^{\mathcal{I}} \cap \mathsf{D}^{\mathcal{I}} \\
(\mathsf{C} \sqcup \mathsf{D})^{\mathcal{I}} &:= \mathsf{C}^{\mathcal{I}} \cup \mathsf{D}^{\mathcal{I}} \\
(\neg \mathsf{C})^{\mathcal{I}} &:= \Delta^{\mathcal{I}} \setminus \mathsf{C}^{\mathcal{I}} \\
(\exists\, \mathsf{R} \,.\, \mathsf{C})^{\mathcal{I}} &:= \{a \in \Delta^{\mathcal{I}} \mid \exists\, b \in \Delta^{\mathcal{I}} : (a,b) \in \mathsf{R}^{\mathcal{I}}, b \in \mathsf{C}^{\mathcal{I}}\} \\
(\forall\, \mathsf{R} \,.\, \mathsf{C})^{\mathcal{I}} &:= \{a \in \Delta^{\mathcal{I}} \mid \forall\, b : (a,b) \in \mathsf{R}^{\mathcal{I}} \Rightarrow b \in \mathsf{C}^{\mathcal{I}}\} \\
(\exists\, \mathsf{u}_1, \ldots, \mathsf{u}_n \,.\, \mathsf{P})^{\mathcal{I}} &:= \{a \in \Delta^{\mathcal{I}} \mid \exists x_1, \ldots, x_n \in \Delta^{\mathcal{D}} : \\
&\qquad (a, x_1) \in \mathsf{u}_1{}^{\mathcal{I}}, \ldots, (a, x_n) \in \mathsf{u}_n{}^{\mathcal{I}}, \\
&\qquad (x_1, \ldots, x_n) \in \mathsf{P}^{\mathcal{I}}\} \\
(\exists\, (\mathsf{u}_1, \ldots, \mathsf{u}_n)(\mathsf{v}_1, \ldots, \mathsf{v}_m) \,.\, \mathsf{P})^{\mathcal{I}} &:= \{(a,b) \in \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}} \mid \\
&\qquad \exists x_1, \ldots, x_n, y_1, \ldots, y_m \in \Delta^{\mathcal{D}} : \\
&\qquad\quad (a, x_1) \in \mathsf{u}_1{}^{\mathcal{I}}, \ldots, (a, x_n) \in \mathsf{u}_n{}^{\mathcal{I}}, \\
&\qquad\quad (b, y_1) \in \mathsf{v}_1{}^{\mathcal{I}}, \ldots, (b, y_m) \in \mathsf{v}_m{}^{\mathcal{I}}, \\
&\qquad\quad (x_1, \ldots, x_n, y_1, \ldots, y_m) \in \mathsf{P}^{\mathcal{I}}\}
\end{aligned}
$$

An interpretation $\mathcal{I}$ is a *model* of a TBox $\mathcal{T}$ iff it satisfies $\mathsf{A}^{\mathcal{I}} \subseteq \mathsf{D}^{\mathcal{I}}$ ($\mathsf{A}^{\mathcal{I}} = \mathsf{D}^{\mathcal{I}}$) for all concept introduction axioms $\mathsf{A} \sqsubseteq \mathsf{D}$ ($\mathsf{A} \doteq \mathsf{D}$) in $\mathcal{T}$ and $\mathsf{S}^{\mathcal{I}} = \mathsf{R}^{\mathcal{I}}$ for all terminological axioms $\mathsf{S} \doteq \mathsf{R}$ (role introductions) in $\mathcal{T}$.

### 3.2.2. The Assertional Language of $\mathcal{ALCRP}(\mathcal{D})$

In the following, the language for representing knowledge about individuals is introduced. An *ABox* $\mathcal{A}$ is a finite set of assertional axioms which are defined as follows:

**Definition 15 (Assertional Axioms, ABox)** Let $O$ be a set of individual names. Furthermore, let $X$ be a set of names for concrete objects ($X \cap O = \emptyset$). If $\mathsf{C}$ is a concept term, $\mathsf{R}$ a role name, $\mathsf{a}, \mathsf{b} \in O$ are individual names and $\mathsf{x}, \mathsf{x}_1, \ldots, \mathsf{x}_n \in X$ are names for concrete objects, then the following expressions are *assertional axioms*:

- $\mathsf{a} : \mathsf{C}$ *(concept assertion)*,
- $(\mathsf{a}, \mathsf{b}) : \mathsf{R}$ *(role assertion)*,
- $(\mathsf{a}, \mathsf{x}) : \mathsf{f}$ *(concrete domain feature assertion)*,
- $(\mathsf{x}_1, \ldots, \mathsf{x}_n) : \mathsf{P}$ *(concrete domain predicate assertion)*.

The interpretation function $\cdot^{\mathcal{I}}$ of the interpretation $\mathcal{I}$ for the concept language can be extended to the assertional language by additionally mapping every individual name from $O$ to a single element $\Delta^{\mathcal{I}}$ (the unique name assumption does not necessarily hold). Concrete objects from $X$ are mapped to elements of $\Delta^{\mathcal{D}}$.

An interpretation satisfies an assertional axiom $\mathsf{a} : \mathsf{C}$ iff $\mathsf{a}^{\mathcal{I}} \in \mathsf{C}^{\mathcal{I}}$, $(\mathsf{a}, \mathsf{b}) : \mathsf{R}$ iff $(\mathsf{a}^{\mathcal{I}}, \mathsf{b}^{\mathcal{I}}) \in \mathsf{R}^{\mathcal{I}}$, $(\mathsf{a}, \mathsf{x}) : \mathsf{f}$ iff $(\mathsf{a}^{\mathcal{I}}, \mathsf{x}^{\mathcal{I}}) \in \mathsf{f}^{\mathcal{I}}$ and $(\mathsf{x}_1, \ldots, \mathsf{x}_n) : \mathsf{P}$ iff $(\mathsf{x}_1{}^{\mathcal{I}}, \ldots, \mathsf{x}_n{}^{\mathcal{I}}) \in \mathsf{P}^{\mathcal{I}}$.

An interpretation $\mathcal{I}$ is a *model* of an ABox $\mathcal{A}$ w.r.t. a TBox $\mathcal{T}$ iff it is a model of $\mathcal{T}$ and furthermore satisfies all assertional axioms in $\mathcal{A}$.

### 3.2.3. Examples for Reasoning with $\mathcal{ALCRP}(\mathcal{D})$

Our second example will demonstrate reasoning with a concrete domain which can represent spatial relations between domain objects. We focus on topological relations known from the RCC theory [31].

Before presenting the example we briefly introduce the concrete domain $\mathcal{RCC}$. We will consider specific spatial objects whose spatial representations are given as polygons. We will show that $\mathcal{RCC}$ provides predicates which can be used to describe qualitative spatial RCC-8 relations as roles between spatial objects.

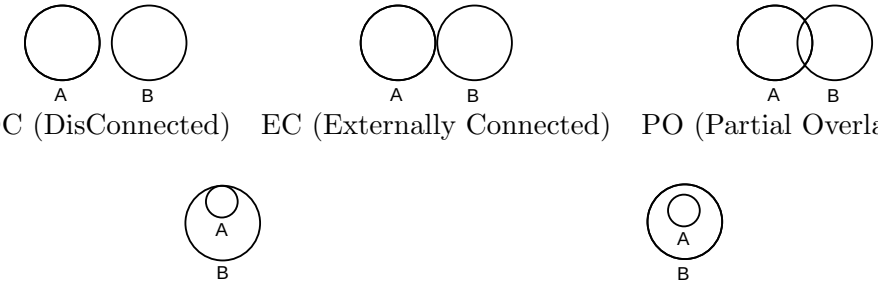
**Definition 16** The concrete domain $\mathcal{RCC}$ is defined w.r.t. the topological space $\langle \mathbb{R}^2, 2^{\mathbb{R}^2} \rangle$. The domain $\Delta^{\mathcal{RCC}}$ contains all non-empty, regular closed subsets of $\mathbb{R}^2$ which are called *regions* for short. The set of predicate names is defined as follows:

- A unary *concrete_domain_top* predicate $\mathsf{is\text{-}region}$ with $\mathsf{is\text{-}region}^{\mathcal{RCC}} = \Delta^{\mathcal{RCC}}$ and its negation $\mathsf{is\text{-}no\text{-}region}$ with $\mathsf{is\text{-}no\text{-}region}^{\mathcal{RCC}} = \emptyset$.
- The 8 basic predicates $\mathsf{dc}$, $\mathsf{ec}$, $\mathsf{po}$, $\mathsf{tpp}$, $\mathsf{ntpp}$, $\mathsf{tppi}$, $\mathsf{ntppi}$ and $\mathsf{eq}$ correspond to the RCC-8 relations. The intuitive semantics of the topological relations from RCC-8 is presented in Figure 3. Due to space restrictions we would like to refer to [16] for a formal definition of the semantics.
- In order to name disjunctions of base relations, we need additional predicates. Unique names for these "disjunction predicates" are enforced by imposing the following canonical order on the basic predicate names: $\mathsf{dc}$, $\mathsf{ec}$, $\mathsf{po}$, $\mathsf{tpp}$, $\mathsf{ntpp}$, $\mathsf{tppi}$, $\mathsf{ntppi}$, $\mathsf{eq}$. For each sequence $p_1, \ldots, p_n$ of basic predicates in canonical order ($n \geq 2$), an additional predicate of arity 2 is defined. The predicate has the name $\mathsf{p_1\text{-}\cdots\text{-}p_n}$ and we have $(r_1, r_2) \in \mathsf{p_1\text{-}\cdots\text{-}p_n}^{\mathcal{RCC}}$ iff $(r_1, r_2) \in \mathsf{p_1}^{\mathcal{RCC}}$ or $\ldots$ or $(r_1, r_2) \in \mathsf{p_n}^{\mathcal{RCC}}$. The predicate $\mathsf{dc\text{-}ec\text{-}po\text{-}tpp\text{-}ntpp\text{-}tppi\text{-}ntppi\text{-}eq}$ is also called $\mathsf{spatially\text{-}related}$.
- A binary predicate $\mathsf{inconsistent\text{-}relation}$ with $\mathsf{inconsistent\text{-}relation}^{\mathcal{RCC}} = \emptyset$ is the negation of $\mathsf{spatially\text{-}related}$.

**Proposition 17** The concrete domain $\mathcal{RCC}$ is admissible.

*Proof.* This is proven in [16]. □

Based on the results presented in [32] we can conclude that there exists always a model whose individuals are polygons which are not necessarily internally connected.




DC (DisConnected) EC (Externally Connected) PO (Partial Overlapping)

TPP (Tangential Proper Part) (NTPP Non-Tangential Proper Part)

Figure 3: Elementary relations between two regions A and B. The inverses of TPP (TPPI) and NTPP (NTPPI) as well as the relation EQ (EQual) are not shown.

With the concrete domain $\mathcal{RCC}$ introduced above, we can now turn to an example where the expressive power of $\mathcal{ALCRP}(\mathcal{D})$ is demonstrated in the context of spatioterminological reasoning. First, we consider a set of role introduction axioms. As an ontological commitment it is assumed that spatial objects (e.g polygons) are associated with abstract objects via the feature has_area.

**related** $\doteq$ $\exists$(has_area)(has_area).spatially-related

**inside** $\doteq$ $\exists$(has_area)(has_area).tpp-ntpp

**inside_i** $\doteq$ $\exists$(has_area)(has_area).tppi-ntppi

**touching** $\doteq$ $\exists$(has_area)(has_area).ec

**overlapping** $\doteq$ $\exists$(has_area)(has_area).po-tpp-ntpp-tppi-ntppi-eq

**spatially_connected** $\doteq$ $\exists$(has_area)(has_area).ec-po-tpp-ntpp-tppi-ntppi-eq

The following concept introduction axioms constitute the TBox of our example for reasoning in $\mathcal{ALCRP}(\mathcal{RCC})$.

**coastal_city** $\doteq$ city $\sqcap$ $\exists$ touching . sea

**country** $\sqsubseteq$ $\forall$ overlapping . $\neg$sea

**sea** $\sqsubseteq$ ocean

A coastal_city is defined as a city which is touched by a sea (in the sense of ocean). Furthermore, the second axiom enforces that countries do not overlap with seas.

The inferential power of $\mathcal{ALCRP}(\mathcal{RCC})$ is explained with an instance problem concerning the following ABox.

$$
\begin{aligned}
\mathsf{loc\_1} &: \mathsf{city} \\
\mathsf{country\_1} &: \mathsf{country} \\
\mathsf{port\_1} &: \mathsf{port} \\
\mathsf{sea\_1} &: \mathsf{sea} \\
(\mathsf{loc\_1}, \mathsf{country\_1}) &: \mathsf{inside} \\
(\mathsf{loc\_1}, \mathsf{port\_1}) &: \mathsf{inside\_i} \\
(\mathsf{port\_1}, \mathsf{sea\_1}) &: \mathsf{touching}
\end{aligned}
$$

We consider a specific instance problem $\mathsf{instance?}(\mathsf{loc\_1}, \mathsf{coastal\_city})$ posed as a query to the description logic system. As indicated above (see Section 3.1.1., the answer of the query is determined by the test whether the ABox becomes inconsistent if the assertion $\mathsf{loc\_1} : \neg\mathsf{coastal\_city}$ is added.

Possible spatial configurations based on ABox information are shown in Figure 4. Considering the definition of inside the topological relation between the city loc_1 and the country country_1 is either tpp (tangential proper part) or ntpp (non-tangential proper part). The basic relations between the city and the port are tppi or ntppi (i for inverse), i.e. the port port_1 is a tangential or a non-tangential proper part of the city. Since the port touches the sea sea_1 (relation ec, externally connected) and, due to the second terminological axiom, a country and a sea cannot overlap (base relations po, tppi, ntppi or eq) only the third configuration in Figure 4 leads to a consistent scenario. However, since, due to the query, it is claimed that $\mathsf{loc\_1} : \neg\mathsf{coastal\_city}$ holds, there must not be a sea touching the city (see the terminological axiom for coastal_city). Hence, the ABox is inconsistent and the answer to the query $\mathsf{instance?}(\mathsf{loc\_1}, \mathsf{coastal\_city})$ is 'yes'.

### 3.3. Decidability Results

From the above it should be evident that ABox consistency checking is at the heart of the reasoning required for information retrieval. We now state the decidability results relevant for our problem scenario.

**Theorem 18** The Abox consistency problem for $\mathcal{ALCNH}_{R^+}$ is decidable.

*Proof.* The proof is given in [17]. □

From the discussion above, it should be clear that once the ABox consistency problem for $\mathcal{ALCNH}_{R^+}$ is shown to be decidable, all inference problems mentioned above can be solved.

In [25] as well as [15] it is shown that, unfortunately, the inference problem of checking the consistency of ABoxes in the "generic" language $\mathcal{ALCRP}(\mathcal{D})$ is undecidable in general. However, in [16] a restricted variant of $\mathcal{ALCRP}(\mathcal{D})$ is described that is indeed decidable if only (syntactically) *restricted* concept terms are used. Thus,

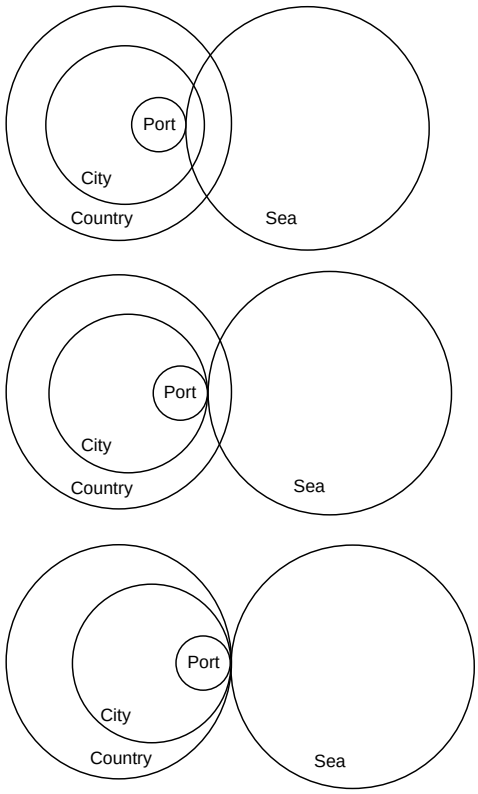



Figure 4: Candidate configurations for ABox individuals (see text).

the above-mentioned $\mathcal{ALCRP}(\mathcal{D})$ inference problems can be decided if only restricted $\mathcal{ALCRP}(\mathcal{D})$ concept terms are admitted.

An unfolded term is in negation normal form if negation is used only for concept names (for details see [16]).

**Definition 19** A concept term $\mathsf{X}$ is called *restricted* w.r.t. a TBox $\mathcal{T}$ iff its equivalent $\mathsf{X}'$ which is unfolded w.r.t. $\mathcal{T}$ and in negation normal form fulfills the following conditions:[4]

(1) For any subconcept term $\mathsf{C}$ of $\mathsf{X}'$ that is of the form $\forall\,\mathsf{R}_1\,.\,\mathsf{D}$ ($\exists\,\mathsf{R}_1\,.\,\mathsf{D}$) where $\mathsf{R}_1$ is a complex role term, $\mathsf{D}$ does not contain any terms of the form $\exists\,\mathsf{R}_2\,.\,\mathsf{E}$ ($\forall\,\mathsf{R}_2\,.\,\mathsf{E}$) where $\mathsf{R}_2$ is also a complex role term.

(2) For any subconcept term $\mathsf{C}$ of $\mathsf{X}'$ that is of the form $\forall\,\mathsf{R}\,.\,\mathsf{D}$ or $\exists\,\mathsf{R}\,.\,\mathsf{D}$ where $\mathsf{R}$ is a complex role term, $\mathsf{D}$ contains only predicate exists restrictions that (i) quantify over attribute chains of length 1 and (ii) are not contained inside any value and exists restrictions that are also contained in $\mathsf{D}$.

A terminology is called restricted iff all concept terms appearing on the right-hand side of terminological axioms in $\mathcal{T}$ are restricted w.r.t. $\mathcal{T}$. An ABox $\mathcal{A}$ is called restricted w.r.t. a TBox $\mathcal{T}$ iff $\mathcal{T}$ is restricted and all concept terms used in $\mathcal{A}$ are restricted w.r.t. the terminology $\mathcal{T}$.

[4]For technical reasons, we assume that a concept term is a subconcept term of itself.

**Theorem 20** The ABox consistency problem for restricted $\mathcal{ALCRP}(\mathcal{D})$ concept terms is decidable if $\mathcal{D}$ is an admissible concrete domain.

*Proof.* The proof is given in [16]. □

**Proposition 21** The set of restricted $\mathcal{ALCRP}(\mathcal{D})$ concept terms is closed under negation.

*Proof.* See [16]. □

As a corollary we can conclude that for $\mathcal{ALCRP}(\mathcal{D})$ all inference problems explained in Section 3.1.1. can be decided if only restricted concept terms are involved. The TBox and the ABox of the example in the previous section contain only restricted concept terms. Therefore the inference problems are decidable.

We have seen that the logics $\mathcal{ALCNH}_{R^+}$ and $\mathcal{ALCRP}(\mathcal{RCC})$ are decidable. Indeed, for both logics, a reasoner has actually been implemented. The ABox description logic system RACE [18] is based on a highly optimized variant of the calculus for $\mathcal{ALCNH}_{R^+}$ [17]. The implementation of the logic $\mathcal{ALCRP}(\mathcal{D})$ is described in [34]. Furthermore, a consistency tester for the concrete domain $\mathcal{RCC}$ has been implemented. Thus the implementation prerequisites for using these languages in practical applications are available.

## 4. DLs as the Basis for Agent-Based Information Systems

In this paper we are particularly interested in information retrieval. In the scenario of the introduction, a specific kind of information retrieval called example-based information retrieval is introduced. For this task, additional inference problems are defined in the next section.

### 4.1. Example-Based Instance Retrieval

In the previous section we have seen that Aboxes can be used to represent knowledge about particular individuals. A common task is to find individuals that satisfy certain conditions. For this purpose, the instance retrieval inference problem is defined.

**Definition 22 (Instance Retrieval)** The *instance retrieval* inference problem is to find all individuals mentioned in an ABox that are an instance of a certain concept term $\mathsf{C}$ w.r.t. a TBox.

The retrieval can be reduced to subsequent instance problems and, therefore, can also be reduced to the ABox consistency problem.

The instance retrieval inference problem can be slightly extended. We can characterize the set of individuals which are to be retrieved not only by a concept term but also by a set of example individuals $\{i_1, \ldots, i_n\}$. We will show that the example individuals can be used to derive a second concept term describing individuals which are "related" to the example individuals.

In example-based instance retrieval the explicit concept term is called a filter $F$. Note that the individuals $\{i_1, \ldots, i_n\}$ need not be instances of $F$.

In order to retrieve individuals "related" to example individuals it is necessary to compute an abstraction representing the commonalities of the example individuals. As "relatedness" of individuals is usually hard to define, we pursue a terminology-based approach. Instead of the individuals, we consider the direct types of the individuals with respect to a TBox. More specifically, we describe each individual of the example-based instance retrieval query by the conjunction of its direct types. Informally speaking, the idea is to compute an abstraction for all direct type conjunctions. The abstraction will represent the commonalities of the example individuals as described by the direct type conjunctions.

We now give a formalization of the commonalities of a set of concepts in terms of the Least Common Subsumer (LCS).

**Definition 23 (Least Common Subsumer)** A concept $\mathsf{C}$ is a least common subsumer of $\mathsf{D1}$ and $\mathsf{D2}$ iff $\mathsf{C}$ subsumes both $\mathsf{D1}$ and $\mathsf{D2}$ and there is no other common subsumer of $\mathsf{D1}$ and $\mathsf{D2}$ that is subsumed by $\mathsf{C}$ (see [10]).

If the language contains an OR operator, the LCS of two concepts is the disjunction of the input concepts. For $\mathcal{ALN}$ with concept introduction axioms (and no GCIs) the LCS of two concepts can be computed as follows [10]. If the LCS operation is applied w.r.t. a TBox, before applying the following LCS function, the arguments must be unfolded. Any concept term can be transformed into an unfolded form by iteratively replacing (or inserting) concept names by their defining terms (for details see [16]).

- $\mathsf{LCS}(\mathsf{C}_{1_1} \sqcap \ldots \sqcap \mathsf{C}_{1_k}, \mathsf{C}_{2_1} \sqcap \ldots \sqcap \mathsf{C}_{2_l}) := \mathsf{LCS}(\mathsf{C}_{1_1}, \mathsf{C}_{2_1}) \sqcap \ldots \sqcap \mathsf{LCS}(\mathsf{C}_{1_k}, \mathsf{C}_{2_l})$
- $\mathsf{LCS}(\forall\, \mathsf{r1} \,.\, \mathsf{C}, \forall\, \mathsf{r2} \,.\, \mathsf{D}) := \mathsf{if}\ \mathsf{r1} = \mathsf{r2}\ \mathsf{then} \forall\, \mathsf{r1} \,.\, \mathsf{LCS}(\mathsf{C}, \mathsf{D})\ \mathsf{else} \top$
- $\mathsf{LCS}(\exists_{\geq n}\, \mathsf{r1}, \exists_{\geq m}\, \mathsf{r2}) := \mathsf{if}\ \mathsf{r1} = \mathsf{r2}\ \mathsf{then}\ (\exists_{\geq min(n,m)}\, \mathsf{r1})\ \mathsf{else} \top$
- $\mathsf{LCS}(\exists_{\leq n}\, \mathsf{r1}, \exists_{\leq m}\, \mathsf{r2}) := \mathsf{if}\ \mathsf{r1} = \mathsf{r2}\ \mathsf{then}\ (\exists_{\leq max(n,m)}\, \mathsf{r1})\ \mathsf{else} \top$

It can be easily verified that the LCS is an associative operation. Using the LCS operation we define a sequence $Q_i$ of query concepts. Let $T_i$ be the conjunction of the direct types of the individual $i_k$. Further, let $Q_0 := \{T_1, \ldots, T_n\}$ be the set of direct type conjunctions for each example individual $i_k$ given as parameter to the example-based instance retrieval operation ($k \in 1..n$). Furthermore, $LCS_i := lfold(\mathrm{LCS}, \top, (Q_{i_1}, \ldots, Q_{i_{n_i}}))$. The function $lfold$ successively applies the first argument, a left-associative function $f$, to the result of the previous application of $f$ (initially $\top$ is used) and the components of the third argument (a sequence). The index $n_i$ is the cardinality of the set $Q_i$. $Q_{i+1}$ is defined as the set of parents of $LCS_i$. Note that there always exists a $LCS_l$ which is equal to $\top$.

**Definition 24 (Example-Based Instance Retrieval)** The *example-based instance retrieval problem* is to find a minimal $i$ such that the set of example individuals

$\{i_1, \ldots, i_n\}$ is a proper subset of the set of individuals $S\_Q_i$ retrieved by the query $instance\_retrieval(F \sqcap Q_{i_1} \sqcap \ldots \sqcap Q_{i_{n_i}})$ or $S\_LCS_i$ retrieved by $instance\_retrieval(F \sqcap LCS_i)$. Once the minimal $i$ is obtained, the result of the example-based instance retrieval operation is $S\_Q_i$ if $\{i_1, \ldots, i_n\}$ is a proper subset of $S\_Q_i$, or $S\_LCS_i$ otherwise. The concept $F$ is the filter concept introduced above.

### 4.2. Multi-Agent Instance Retrieval

Compared to $\mathcal{ALCNH}_{R^+}$ the description logic part of $\mathcal{ALCRP}(\mathcal{D})$ is less expressive. Thus, the question arises if, for instance, the predicate exists restriction construct of $\mathcal{ALCRP}(\mathcal{D})$ (see above) could be added to $\mathcal{ALCNH}_{R^+}$. Unfortunately, this leads to decidability problems.

**Theorem 25 (Undecidability of $\mathcal{ALCNH}_{R^+}(\mathcal{D})$)** The concept consistency problem for $\mathcal{ALCNH}_{R^+}(\mathcal{D})$ is not decidable.

*Proof.* The proof is given in [19]. □

The undecidability result shows that, for instance, reasoning about conceptual and spatial information in an expressive language that combines transitive roles and role hierarchies with concrete domains cannot be achieved in the general case.

It is possible to define additional syntactic restrictions. A restriction of predicate exists restrictions to features only (and no feature chains) has been investigated in in [19]. It is is shown that the ABox consistency problem for the logic $\mathcal{ALCNH}_{R^+}(\mathcal{D})^-$ is decidable.

However, it is currently unknown whether roles define on the basis of concrete domain predicates can be integrated into $\mathcal{ALCNH}_{R^+}(\mathcal{D})^-$. Therefore, we pursue another approach. Rather than directly combining the logics presented in this section we introduce an architecture that uses agents as separate "competence centers." We will describe a scenario where each agent employs a sound and complete inference procedure. Although a calculus for the combination of $\mathcal{ALCNH}_{R^+}$ and $\mathcal{ALCRP}(\mathcal{D})$ must be incomplete, abstraction processes for data (ABoxes) communicated between agents and specialization processes for queries ensure that at least a sound and terminating combination of inference algorithms can be achieved in a distributed architecture.

Agents comprising different knowledge bases expressed in different description logics cooperate for information retrieval. The approach presented in this section presumes that one of the agents plays the part of a broker, the others, called specialists, supply information to the broker. The broker receives queries from a user, communicates with specialists, and delivers answers to the user. Communication with the user is performed using the broker's terminology. To be able to communicate with the other agents, the broker has knowledge about the concept and role names of each agent, and how they relate to the broker's terminology (interschema axioms [9]). The main task of the broker is to transform queries into the terminology of another agent and transform answers back into the broker's terminology. In the following, this will be described more precisely.

### 4.2.1. Inference Problems w.r.t. Namespaces

We have seen that in order to use a description logic in an application, a set of atomic concepts and roles tailored to the application must be specified. Relationships between the atomic concepts or roles can be defined with axioms. To distinguish between different knowledge bases, we define the notion of a namespace.

**Definition 26 (Namespace)** A namespace $\mathsf{n}$ is a set of atomic concepts or atomic roles with a common prefix $\mathsf{n}$: where $\mathsf{n}$ is the name of the namespace.

**Definition 27 (Concept and Role from a Namespace)** A concept is called *a concept from a namespace* $\mathsf{n}$ iff all atomic subconcepts have the prefix $\mathsf{n}$:. A role is called *a role from a namespace* $\mathsf{n}$ iff it has a prefix $\mathsf{n}$:.

Using the notion of a namespace, we can now specify the special role of the broker agent. Its knowledge base contains atomic concepts and roles from multiple namespaces (with different prefixes). Thus the broker can perform inference services w.r.t. to different namespaces. For instance, the broker can compute the *parents* of a (not necessarily atomic) concept $\mathsf{C}$ w.r.t. a namespace $\mathsf{n}$ and a TBox $\mathcal{T}$. The result is the set of most-specific atomic concepts in the namespace $\mathsf{n}$ that subsume $\mathsf{C}$. The set of *children* w.r.t. to a namespace and the *direct types* of an individual w.r.t. a certain namespace are defined analogously.

### 4.2.2. Namespace Transformations for Concepts, Roles and ABoxes

Let us consider an instance checking task now, and how the broker may invoke a specialist. The query consists of a concept term, an individual name and an ABox in the broker's terminology. The task is to check whether the individual is an instance of the concept term. If the broker can prove w.r.t. its own knowledge base either that the individual is an instance of the concept term or that it is not, the task is solved (but see the section on inconsistent queries below). If the broker's knowledge is inconclusive, a specialist may help. Hence the query must be transformed into an approximate query with concepts and roles only in the namespace of the specialist. The key idea is to transform the query such that the ABox is abstracted and the concept term is refined (or specialized). Due to space restrictions we can give only a sketch of the algorithms. Furthermore, it might be necessary to ensure that abstractions and refinements fulfill certain restrictedness criteria (see Section 3.3.).

**Definition 28 (ABox Abstraction)** An ABox $\mathcal{A}'$ is an *abstraction* of an ABox $\mathcal{A}$ iff $\mathcal{A} \models \mathcal{A}'$ holds.

We use the following heuristics to compute an ABox abstraction. For each role found in a role assertion of an ABox the most-specific superrole of the namespace of the destination agent is inserted. Note that the most-specific superrole in the namespace of the destination agent may also be a synonym of the role being transformed. If the most-specific superrole is not unique, in the namespace of the consulted agent a new

role is dynamically added with appropriate inclusion axioms such that the new role is a subrole of the set of most-specific superroles. If no superrole exists in the TBox of a certain destination agent, then *agent_name*:$\mathsf{related}$ is used. The role $\mathsf{related}$ is assumed to be a superrole of all atomic roles of an agent. Similar transformations are employed for the atomic concepts used in concept assertions. Either synonyms or the conjunction of the parents w.r.t. the namespace of the destination agent are inserted. Note that in the worst case $\top$ will be returned as an abstraction of a certain concept.

**Definition 29 (Concept Refinement)** A concept $\mathsf{C}'$ is a *refinement* of a concept $\mathsf{C}$ iff $\mathsf{C}$ subsumes $\mathsf{C}'$.

There are different strategies for computing concept refinements. First, in order to refine a query concept $C_0$, the children w.r.t. the namespace of the destination agent are computed. Given the children $C_1, \ldots, C_n$ of a concept $C_0$, a refinement is the disjunction $C_1 \sqcup \ldots \sqcup C_n$. If only the bottom concept $\bot$ is returned as a child of $C_0$, a second strategy is employed. A refinement can be computed w.r.t. the form of the concept. In a similar way as in the abstraction process of the ABox, the roles and the concept terms in the query concepts are transformed. For brevity, let us consider an existential restriction $\exists\,\mathsf{R}\,.\,\mathsf{C}$ as an example. The transformation is a concept $\exists\,\mathsf{R}'\,.\,\mathsf{C}'$ where $\mathsf{R}'$ is the most-general subrole of $\mathsf{R}$ in the namespace of the destination agent and $\mathsf{C}'$ is the result of the (recursive) transformation of $\mathsf{C}$. For atomic concepts, the transformation is the disjunction of the children w.r.t. the namespace of the destination agent (see the first strategy). The refinement or rewriting of concepts using terminologies has also been investigated, for instance, in [3] and [5].

### 4.2.3. Broker-Based Query Answering

Let us consider now how the answers of different agents can be combined from a logical perspective.

We will first take a principled view of this problem. Let us assume that we have two TBoxes, $\mathcal{T}_1$ and $\mathcal{T}_2$, representing the knowledge of the agents $A$ and $B$, respectively. For instance, let $\mathcal{T}_1$ be an $\mathcal{ALCNH}_{R^+}$ TBox and $\mathcal{T}_2$ be an $\mathcal{ALCRP}(\mathcal{D})$ TBox. Considering the results of Section 4.2. we know that the combination of $\mathcal{ALCNH}_{R^+}$ and $\mathcal{ALCRP}(\mathcal{D})$ is undecidable. Hence, in general, we know that there is no sound and complete (and terminating) calculus for answering a query posed to two knowledge bases. Nevertheless, let us consider an instance checking problem $instance?(i, C)$ w.r.t. the knowledge base $(\mathcal{T}_1 \cup \mathcal{T}_2, \mathcal{A})$. Furthermore, let us assume that $\mathcal{A}$ is an ABox of $\mathcal{T}_1$ and that $\mathcal{A}$ is not inconsistent. The idea is to address the query - or a suitable transformation thereof - to each knowledge base separately. If the answer to the instance checking problem $instance?(i, C)$ w.r.t. $(\mathcal{T}_1, \mathcal{A})$ is 'yes', then it is obvious that the answer w.r.t. $(\mathcal{T}_1 \cup \mathcal{T}_2, \mathcal{A})$ is 'yes' as well. In order to address the query to $\mathcal{T}_2$ we may have to transform it since we assume that $\mathcal{A}$ is a $\mathcal{T}_1$ ABox. For instance, $\mathcal{A}$ may contain number restrictions which are no $\mathcal{T}_2$ concept terms. If we transform the instance checking problem such that $instance?(i, C')$ w.r.t. $(\mathcal{T}_2, \mathcal{A}')$ is solved, with $\mathcal{A}'$

an abstraction and $C'$ a refinement, it can be easily seen that we can exploit the answer for the original query: If the answer to the subproblem $instance?(i, C')$ w.r.t. $(\mathcal{T}_2, \mathcal{A}')$ is 'yes' then the answer to $instance?(i, C)$ w.r.t. $(\mathcal{T}_1 \cup \mathcal{T}_2, \mathcal{A})$ is 'yes' as well.

If both terminologies return 'no', we cannot be sure of the overall result because of undecidability. But we can attempt to solve the inference problem $instance?(i, \neg C)$, again by addressing each knowledge base separately. If the answer to the complementary query is 'yes' then the answer to the original query $instance?(i, C)$ is 'no' for sure.

Although some limitations are inevitable, we now have the basis for a sound algorithm in an arbitrary agent scenario. Each agent has a sound and complete (and terminating) sub-algorithm for the inference tasks in question. There is one specific agent, the broker, which poses inference problems to other agents. For this task the broker needs a TBox that is a close approximation of the TBoxes of the other agents.

We now put things together and describe how an instance checking query can be dealt with in a multiagent scenario. The broker first tries to answer the query himself using his own ABox. If the answer is negative, the broker consults an agent whose namespace contains the individual of the query. The concept of the query must be specialized and the broker's ABox fragment of the individual (if any) must be abstracted in order to preserve correctness. If the answer of the specialist is positive, this answer is returned by the broker. In the case of a negative answer, we assume that the specialist returns his ABox fragment in which the individual in question is mentioned. The broker transforms this information into his terminology and can now consult other specialist agents using the enriched ABox information about the individual. In each case, the atomic concepts and roles are translated into the specialist's namespace while preserving correctness by ABox abstraction and concept refinement. If one of the specialists returns 'yes' then the broker returns 'yes', otherwise the answer is undecided. In this case, the broker may try to answer the query with the negated concept. If he (or one of the specialists) succeeds, the answer to the original query is 'no', otherwise it remains undecided.

A broker delegates an example-based instance retrieval task to a specialist agent whose name is determined by the namespace of the individuals mentioned in the example-based instance retrieval query. As a restriction, all individuals must be from the same namespace. The filter concept is a concept from the namespace of the broker. Therefore, it has to be transformed to be "understandable" by the consulted specialist agent. The transformed filter concept is defined to be a concept refinement w.r.t. the namespace of the consulted agent.

## 5. An Extended Application Example

As we have discussed in the introduction, the motivating scenario for the agent-based architecture which we investigate in this article is an information retrieval application. In particular, we focus on a TV-Assistant whose main task is to provide personalized TV-programs. The user can ask for a personalized TV-program by (i) providing a

time window and (ii) examples of broadcasts which characterize his interests. The TV-Assistant will then retrieve broadcasts of that time window (the so-called basic selection) and mark broadcasts which are of special interest to the user according to his examples (his personal selection). This can be done, for example, by highlighting as shown in Figure 1.

As a second task, the TV-Assistant has to select advertisements to be inserted into the personalized program display. To accomplish this, advertisements are associated with conceptual descriptions, so-called trigger concepts, of the types of broadcasts for which the advertisements should be displayed. So if a broadcast contained in the personal selection of a user turns out to be an instance of a trigger concept, the associated advertisement will be shown. In our application scenario we assume that the TV-Assistant makes use of the services of a multiagent information system as shown in Figure 2).

At first, the basic program selection is retrieved by posing an instance retrieval query to the TV-Broker Agent with the time window expressed as the conceptual constraint. The TV-Broker Agent forwards this query to the Program Agent and receives the basic program selection.

As a second step, the personal program selection is obtained by posing an example-based instance retrieval query to the TV-Broker Agent with the examples provided by the user and the time window as a filter concept. The answer is a set of broadcasts (the personal selection) associated with additional information, for example the actors or the main location in the case of a movie.

Now the advertisements have to be determined based upon the trigger concepts which are maintained by the TV-Assistant. For each broadcast of the personal selection the TV-Broker Agent is asked whether the broadcast is an instance of a trigger concept. In our scenario, there are two agents besides the TV-Broker Agent which can possibly solve the instance checking problem. At first, the Program Agent is asked. When his answer is 'no', the broker turns to a specialist which in our case is a GIS Agent with spatial-reasoning power.

We will now present the agents in detail and then discuss several examples.

### 5.1. The Agents of the Application Example

The Program Agent uses the language $\mathcal{ALN}$ with concept introduction axioms only (i.e. non-cyclic terminological axioms with the additional condition that each terminological axiom is used only once on the left-hand side of the axioms). A prototype system for the Program Agent [29] has been implemented with the knowledge representation system CLASSIC [6] which provides an optimized ABox implementation for the language $\mathcal{ALN}$ (actually, CLASSIC supports a slightly more expressive DL). In this article we present only a subset of the implemented knowledge base. We assume that the TBox of the Program Agent contains the terminological axioms explained in Section 3.1.2.

The Program Agent will perform the example-based instance retrieval procedure ex-

plained in Section 4.1. To this end, we will extend the domain model with definitions for movie. New concepts sailing_ship and titanic are defined as subconcepts of ship.[5] In addition, we assume that soldier and pirate are declared as subconcepts of person. Furthermore, the domain model is extended with concepts for specific movies. Important roles for movies are has_main_character and has_main_location. For the examples we use the following terminological axioms.

$$
\begin{aligned}
&\textbf{soldier} \sqsubseteq \mathsf{person}\\
&\textbf{pirate} \sqsubseteq \mathsf{person}\\
&\textbf{sailing\_ship} \sqsubseteq \mathsf{ship}\\
&\textbf{titanic} \sqsubseteq \mathsf{ship}\\
&\textbf{pirate\_movie} \sqsubseteq\\
&\quad \mathsf{movie} \sqcap\\
&\quad \forall\, \mathsf{has\_main\_character} \,.\, (\mathsf{pirate} \sqcap \mathsf{captain}) \sqcap\\
&\quad \forall\, \mathsf{has\_main\_location} \,.\, \mathsf{sailing\_ship}\\
&\textbf{titanic\_movie} \sqsubseteq\\
&\quad \mathsf{movie} \sqcap\\
&\quad \forall\, \mathsf{has\_main\_character} \,.\, \mathsf{captain} \sqcap\\
&\quad \forall\, \mathsf{has\_main\_location} \,.\, \mathsf{titanic}\\
&\textbf{action\_movie} \sqsubseteq\\
&\quad \mathsf{movie} \sqcap\\
&\quad \forall\, \mathsf{has\_main\_character} \,.\, \mathsf{action\_hero}
\end{aligned}
$$

The structure shown in the ABox below represents a small excerpt of the domain model for objects (ABox) in our TV-Assistant application.

$$
\begin{aligned}
\mathsf{movie\_1} &: \mathsf{movie} \sqcap\\
&\quad \exists_{\leq 1}\, \mathsf{has\_main\_character} \sqcap\\
&\quad \exists_{\leq 1}\, \mathsf{has\_main\_location}\\
\mathsf{hornblower} &: \mathsf{captain} \sqcap \mathsf{soldier}\\
\mathsf{lydia} &: \mathsf{sailing\_ship}\\
(\mathsf{movie\_1}, \mathsf{hornblower}) &: \mathsf{has\_main\_character}\\
(\mathsf{movie\_1}, \mathsf{lydia}) &: \mathsf{has\_main\_location}\\
\mathsf{movie\_2} &: \mathsf{pirate\_movie}\\
\mathsf{movie\_3} &: \mathsf{titanic\_movie}
\end{aligned}
$$

[5] We model titanic as a concept rather than as an instance because different individual ships might carry this name.

For our second example we assume that the Program Agent has detailed information about the James Bond movie the_world_is_not_enough_1:

$$\begin{aligned}
\mathsf{the_world_is_not_enough_1} &: \mathsf{action_movie} \\
\mathsf{james_bond_1} &: \mathsf{action_hero} \\
\mathsf{loc_1} &: \mathsf{baku} \\
\mathsf{loc_2} &: \mathsf{london} \\
\mathsf{country_1} &: \mathsf{azerbaijan} \\
\mathsf{continent_1} &: \mathsf{asia} \\
\mathsf{port_1} &: \mathsf{port} \\
\mathsf{sea_1} &: \mathsf{caspian_sea} \\
(\mathsf{the_world_is_not_enough_1}, \mathsf{james_bond_1}) &: \mathsf{has_main_character} \\
(\mathsf{the_world_is_not_enough_1}, \mathsf{loc_1}) &: \mathsf{has_main_location} \\
(\mathsf{the_world_is_not_enough_1}, \mathsf{loc_2}) &: \mathsf{has_main_location} \\
(\mathsf{loc_1}, \mathsf{country_1}) &: \mathsf{capital_of} \\
(\mathsf{loc_1}, \mathsf{port_1}) &: \mathsf{has_port} \\
(\mathsf{port_1}, \mathsf{sea_1}) &: \mathsf{located_at} \\
(\mathsf{country_1}, \mathsf{continent_1}) &: \mathsf{located_on_continent}
\end{aligned}$$

The GIS Agent uses the description logic $\mathcal{ALCRP}(\mathcal{RCC})$. We assume that the TBox of the GIS Agent contains the role and concept introduction axioms introduced in Section 3.2.3. Before the role of the GIS Agent in our scenario can be understood we first have to discuss the inferences of the central agent, the TV-Broker Agent.

The TV-Broker Agent uses the description logic $\mathcal{ALCNH}_{R^+}$. We assume that all atomic concepts and roles of the Program Agent and the GIS Agent as well as the corresponding terminological axioms are also available in the TBox of the TV-Broker Agent. Atomic concepts and roles "imported" from other agents are indicated by a prefix. We use prefixes pa, ga and ba for the Program Agent, the GIS Agent and the TV-Broker Agent, respectively. Queries to the TV-Broker Agent and trigger concepts use atomic concepts and roles with prefix ba only.

The first part of the TBox of the TV-Broker Agent specifies the relationships between the roles used by the Program Agent and the roles used by the GIS Agent in terms of terminological axioms:

$$\begin{aligned}
\mathsf{pa{:}\textbf{capital_of}} &\sqsubseteq \mathsf{ga{:}inside} \\
\mathsf{pa{:}\textbf{has_port}} &\sqsubseteq \mathsf{ga{:}inside_i} \\
\mathsf{pa{:}\textbf{located_at}} &\sqsubseteq \mathsf{ga{:}touching}
\end{aligned}$$

$$\textbf{pa}\text{:}\textbf{located\_on\_continent} \sqsubseteq \text{ga:inside}$$
$$\textbf{pa}\text{:}\textbf{has\_main\_location} \sqsubseteq \text{ga:inside}$$
$$\textbf{ga}\text{:}\textbf{spatially\_connected} \sqsubseteq \text{ba:spatially\_connected}$$

Within the description logic $\mathcal{ALCN}\mathcal{H}_{R^+}$, the relationships are manifested using role inclusion axioms. Furthermore, in order to approximate the semantics of topological RCC relations, ga:inside and ga:inside_i are declared to be transitive roles. Obviously ga:touching is not a transitive role.

For each agent acquaintance, axioms indicating the direct subsumption relationships (parents and children) of the atomic concepts which are imported from the agent are added automatically to the TBox of the TV-Broker Agent. For instance, the role axiom

$$\textbf{ga}\text{:}\textbf{inside} \sqsubseteq \text{ga:spatially\_connected}$$

is added to the TBox of the TV-Broker Agent. Additional concept inclusion axioms are employed to relate the concept terms of different TBoxes. These axioms cannot be set up automatically but have to be modeled by a system engineer.

$$\textbf{pa}\text{:}\textbf{baku} \sqsubseteq \text{ga:city}$$
$$\textbf{pa}\text{:}\textbf{london} \sqsubseteq \text{ga:city}$$
$$\textbf{pa}\text{:}\textbf{azerbaijan} \sqsubseteq \text{ga:country}$$
$$\textbf{pa}\text{:}\textbf{port} \doteq \text{ga:port}$$
$$\textbf{pa}\text{:}\textbf{caspian\_sea} \sqsubseteq \text{ga:sea}$$
$$\textbf{ga}\text{:}\textbf{coastal\_city} \sqsubseteq \text{ba:coastal\_city}$$
$$\textbf{ga}\text{:}\textbf{coastal\_city} \doteq \text{ga:city} \sqcap \exists\, \text{ga:touching} \,.\, \text{ga:sea}$$
$$\textbf{ga}\text{:}\textbf{country} \sqsubseteq \forall\, \text{ga:overlapping} \,.\, \neg\text{ga:sea}$$
$$\textbf{ba}\text{:}\textbf{asian\_city} \doteq \text{pa:city} \sqcap \exists\, \text{ga:inside} \,.\, \text{pa:asia}$$

Note that generalized concept inclusion axioms are used as well. In addition, all axioms from Section 3.2.3. for representing the knowledge of the GIS Agent are included into the TBox of the TV-Broker Agent (with appropriate prefixes).

### 5.2. Reasoning Examples

As a first example, let us assume that movie_2 and movie_3 and the filter concept C are used in an example-based query posed to the Program Agent.

$$\text{example\_base\_instance\_retrieval}(\{\text{movie\_2}, \text{movie\_3}\}, \text{C})$$

For answering this query, the LCS operation is applied to the (unfolded) direct types of both movies (see Section 4.1.) and returns the following concept:

$$\mathsf{movie} \sqcap \forall\, \mathsf{has\_main\_character} \,.\, \mathsf{captain} \sqcap \forall\, \mathsf{has\_main\_location} \,.\, \mathsf{ship}$$

We can see that an abstraction of the original movies has been computed. From pirate_movie the concept sailing_ship has been abstracted to ship and from titanic_movie the concept titanic has been abstracted to ship as well. The resulting LCS concept presented above is used as a query for retrieving instances from the ABox of the Program Agent. The results to this query are further restricted with the time window filter concept.

In our example the movie movie_1 is an instance of the LCS concept and, therefore, it is returned as an answer (possibly among others).

As a second example, we assume that advertisements for *cruises* and *trips to asian cities* are to be associated with appropriate broadcasts. To trigger the advertisements we consider the concept $\exists\, \mathsf{ba{:}spatially\_connected} \,.\, \mathsf{ba{:}coastal\_city}$ and the concept $\exists\, \mathsf{ba{:}spatially\_connected} \,.\, \mathsf{ba{:}asian\_city}$, which might be set up in the interest of a travel agency. If a retrieval result (e.g. the movie the_world_is_not_enough_1) is an instance of a trigger concept, the associated role fillers for ba:spatially_connected that are instances of ba:coastal_city or ba:asian_city are examined to find cruises or trips offered by the travel agency. The idea is, of course, that after viewing the movie, people might be inclined to book a cruise to the main location of the movie.

The TV-Broker can prove that the_world_is_not_enough_1 is an instance of the concept $\exists\, \mathsf{ba{:}spatially\_connected} \,.\, \mathsf{ba{:}asian\_city}$ because ga:inside is declared as a transitive role in the knowledge base of the TV-Broker Agent. Thus, there is no need to consult another agent.

Unfortunately, given the ABox associated with the_world_is_not_enough_1 (see above) the TV-Broker Agent cannot prove that the movie is an instance of the concept $\exists\, \mathsf{ba{:}spatially\_connected} \,.\, \mathsf{ba{:}coastal\_city}$. It cannot even prove that it is an instance of $\neg\exists\, \mathsf{ba{:}spatially\_connected} \,.\, \mathsf{ba{:}coastal\_city}$.

In our scenario, the TV-Broker Agent therefore asks the GIS Agent to check whether the_world_is_not_enough_1 is an instance of $\exists\, \mathsf{ga{:}spatially\_connected} \,.\, \mathsf{ga{:}coastal\_city}$. After transforming the ABox which the TV-Broker Agent has received from the Program Agent, the following ABox is delegated to the GIS Agent.

$$\begin{aligned} \mathsf{the\_world\_is\_not\_enough\_1} &: \top \\ \mathsf{james\_bond\_1} &: \top \\ \mathsf{loc\_1} &: \mathsf{ga{:}city} \\ \mathsf{loc\_2} &: \mathsf{ga{:}city} \\ \mathsf{country\_1} &: \mathsf{ga{:}country} \end{aligned}$$

$$
\begin{aligned}
\mathsf{port\_1} &: \mathsf{ga{:}port} \\
\mathsf{sea\_1} &: \mathsf{ga{:}sea} \\
(\mathsf{the\_world\_is\_not\_enough\_1}, \mathsf{james\_bond\_1}) &: \mathsf{ga{:}related} \\
(\mathsf{the\_world\_is\_not\_enough\_1}, \mathsf{loc\_1}) &: \mathsf{ga{:}inside} \\
(\mathsf{the\_world\_is\_not\_enough\_1}, \mathsf{loc\_2}) &: \mathsf{ga{:}inside} \\
(\mathsf{loc\_1}, \mathsf{country\_1}) &: \mathsf{ga{:}inside} \\
(\mathsf{loc\_1}, \mathsf{port\_1}) &: \mathsf{ga{:}inside\_i} \\
(\mathsf{port\_1}, \mathsf{sea\_1}) &: \mathsf{ga{:}touching} \\
(\mathsf{country\_1}, \mathsf{continent\_1}) &: \mathsf{ga{:}inside}
\end{aligned}
$$

Considering the example presented in Section 3.2.3. it can be easily verified, that the answer of the GIS Agent to the query is 'yes' because ga:spatially_connected is a superrole of ga:inside. In fact, the third configuration shown in Figure 4 qualitatively describes exactly the relation of Baku and Azerbaijan (see Figure 5).

Once the TV-Broker Agent knows the result of the query, it adds the result to the ABox originally sent to the GIS Agent, in our example the assertional axiom $\mathsf{the\_world\_is\_not\_enough\_1} : \exists\, \mathsf{ga{:}spatially\_connected}\,.\,\mathsf{ga{:}coastal\_city}$. Now the TV-Broker software can be instructed to insert specific travel agency commercials associated with the trigger concepts (in this case e.g. cruises).

Note that, in general, combining knowledge of independent knowledge bases may uncover inconsistencies. For example, the ABox returned by the GIS Agent could prove inconsistent with the ABox of the TV-Broker Agent. In this case no meaningful answer can be supplied.

The example in this section demonstrates the combined expressive power of different DLs. $\mathcal{ALCNH}_{R^+}$ provides generalized concept inclusions, role hierarchies and transitive roles which are needed to represent much of the knowledge required in the application domain. However, some of the ontological interdependencies cannot be captured due to undecidability results. Using the formalism $\mathcal{ALCRP}(\mathcal{RCC})$ we were able to include ontological interdependencies concerning conceptual and spatial knowledge.

It would be possible to extend the language $\mathcal{ALCNH}_{R^+}$ with inverse roles and so-called qualified number restrictions (e.g. [21]) in order to better approximate the knowledge of the specialists (in our case the GIS Agent) in the TV-Broker Agent. However, developing an optimized ABox reasoner implementation for the extended logic is a difficult task and subject to further research. In any case, due to the undecidability result, it would be only an approximation. The agent architecture proposed in this article provides an organized way to cope with the undecidability and incompleteness problems resulting from the combination of different expressive representation languages.


Figure 5: Map indicating the position of Baku, the capital of Azerbaijan.

## 6. Related Work

Information retrieval in a distributed context is a commercially very interesting research topic. There exists a vast amount of scientific contributions and it is hardly possible to cover at least a small subset. Each of the different approaches has its own pros and cons. Here, we focus on related work concerning information retrieval in the context of description logics rather than on work based on database theory and data structure conversion (e.g. the TSIMMIS system [14]) or other knowledge representation approaches (e.g. [13]).

An early work about the application of description logics for information retrieval purposes is [27]. While some authors focus on multi-valued logic in order to capture the notion of "relevance" (e.g. [28]) most contributions rely on a standard semantics. For instance, the Information Manifold project [24] has extended the CLASSIC description logic [6] with so-called conjunctive queries in order to provide a more expressive query language. Solutions for query refinement based on defaults have been developed in [23].

Recently, conjunctive queries for even more expressive description logics such as $\mathcal{ALCNH}_{R^+}$ have been considered. Reasoning about conjunctive queries has been formalized as ABox inclusion which, in turn, can be reduced to ABox consistency [20]. For $\mathcal{ALCRP}(\mathcal{D})$ the reduction of ABox inclusion to ABox consistency is described in [30]. It would be interesting to add conjunctive queries to more expressive description logics such as $\mathcal{ALCNH}_{R^+}$ but (efficient) algorithms for instance retrieval are still an active research area.

The FindUr approach [26] also relies on the CLASSIC system. FindUr uses so-called

ontologies for supporting Web browsing and search. FindUr focuses on the retrieval of Web pages which are annotated with ontological notions. Agent communication in a description logic context has been considered by [22]. In addition and complementary to our approach, game theory is used to control the "activity" of agents.

Schema integration and inter-schema knowledge modeling has been investigated by [9]. In contrast to the approach presented in this article, [9] does not rely on the assumption that the domains of different agents are identical. As a consequence, the notion of intensional inclusion of different concepts is defined (rather than extension inclusions with GCIs). If desired, this could also be considered in our agent scenario. The decomposition of queries in a multidatabase scenario has been considered by [8]. Ideas taken from this context can also be applied in an agent scenario. Newest results on information integration with a description logic capturing the expressiveness of entity-relationship models are described in [7].

## 7. Conclusion

Agent-oriented problem solving has been analyzed from a formal knowledge representation point of view. Based on interschema knowledge a central agent, called broker, transforms inference problems such that they can be delegated to other agents preserving at least a sound overall inference algorithm. We have discussed examples involving instance retrieval and instance checking.

In the first example the motivation for delegation was to employ an agent which uses a less expressive description logic (the Program Agent) such that inferences can be computed more efficiently. In the second example an agent based on a description logic with different expressive power is employed for dealing with an instance checking problem that cannot be solved w.r.t. the knowledge represented by the broker. We have seen that the combined description logic is undecidable in general. Although the abstraction of the ABox and the refinement of the query concept as proposed in this contribution yields a sound inference algorithm based on delegation many inferences will be lost if the abstraction is too general and the refinement is too specific. Therefore, the knowledge of the broker must be a close approximation of the knowledge represented by the consulted specialist. The examples indicate that $\mathcal{ALCNH}_{R^+}$ is well-suited as a representation language for a broker. On the one hand, the logic is expressive enough to approximate the knowledge of other agents. On the other hand, with RACE there exists an implementation which guarantees quite encouraging average-case performance for practical reasoning.

The deficiencies of the approach have been indicated as well. As the combined language $\mathcal{ALCNH}_{R^+}$ and $\mathcal{ALCRP}(\mathcal{D})$ is not decidable in general, it is not possible to check whether any given input ABox is inconsistent w.r.t. the combined knowledge of the overall agent system. Thus, cases where query answering is not very useful due to an inconsistent input ABox might remain undetected. Although a specialist agent might conclude that the abstracted ABox is inconsistent, there are cases where the abstraction is consistent whereas the original is not. An idea to circumvent this

problem in some cases might be to compute a refinement of the input ABox and to let the broker refuse query answering if one of the specialist acquaintances can prove that the refinement is inconsistent. Details of this approach have to be investigated in future work. Furthermore, the development of advanced strategies and heuristics for computing suitable ABox abstractions and concept refinement in a practical scenario is also subject to future investigations.

Implicit knowledge plays an important role in agent-based communication because, for expressive representation languages, there exists no "canonical form" that can be used as format for knowledge interchange. Comparisons between different representation structures have to be computed on a semantical basis. The work presented in this chapter discusses examples in the context of spatioterminological reasoning. Apparently, the key to adequate domain knowledge modeling is not only the definition of many ontological notions with class-subclass or part-whole relations. Instead, representation formalisms that capture the semantics of spatial object are required in order to avoid unintended models. The topological relations we have discussed in this chapter are part of the whole story. Obviously, reasoning facilities for spatial knowledge must be augmented with reasoning techniques for temporal knowledge (see [16] for a first account in the context of spatiotemporal terminological reasoning). Whether the combination of spatial and temporal terminological reasoning can be adequately exploited in the agent-oriented scenario that we have investigated in the chapter is subject to future research.

**Acknowledgments**

The development of the description logic $\mathcal{ALCRP}(\mathcal{D})$ and the concrete domain $\mathcal{RCC}$ is joint work with Carsten Lutz, now at RWTH Aachen.

**References**

[1] F. Baader. Logic-based knowledge representation. In M.J. Wooldridge and M. Veloso, editors, *Artificial Intelligence Today, Recent Trends and Developments*, number 1600 in Lecture Notes in Computer Science, pages 13–41. Springer Verlag, 1999.

[2] F. Baader and P. Hanschke. A scheme for integrating concrete domains into concept languages. In *Twelfth International Conference on Artificial Intelligence, Darling Harbour, Sydney, Australia, Aug. 24-30, 1991*, pages 452–457, August 1991.

[3] F. Baader, R. Küsters, and R. Molitor. Rewriting concepts using terminologies. In A.G. Cohn, F. Giunchiglia, and B. Selman, editors, *Proceedings of the Seventh International Conference on Knowledge Representation and Reasoning (KR2000)*, pages 297–308, San Francisco, CA, 2000. Morgan Kaufmann Publishers.

[4] F. Baader and U. Sattler. Tableau algorithms for description logics. In R. Dyckhoff, editor, *Proceedings of the International Conference on Automated Reasoning with Tableaux and Related Methods (Tableaux 2000)*, volume 1847 of *Lecture Notes in Artificial Intelligence*, pages 1–18, St Andrews, Scotland, UK, 2000. Springer-Verlag.

[5] L. Badea and S.H. Niehuys-Cheng. Refining concepts in description logics. In F. Baader et al., editor, *Proceedings of the International Workshop on Description Logics (DL'2000), August 17 - August 19, 2000, Aachen, Germany*, pages 31–44, August 2000.

[6] R.J. Brachman, D.L. McGuinness, P.F. Patel-Schneider, L.A. Reswnick, and A. Borgida. Living with CLASSIC: When and how to use a KL-ONE-like language. In J.F. Sowa, editor, *Principles of Semantic Networks: Explorations in the Representation of Knowledge*, pages 401–456. Morgan Kaufmann Publishers, San Mateo, 1991.

[7] D. Calvanese, G. De Giacomo, M. Lenzerini, D. Nardi, and R. Rosati. Description logic framework for information integration. In Cohn et al. [11], pages 2–13.

[8] J. Cardiff, T. Catarci, and G. Santucci. Exploitation of interschema knowledge in a multidatabase system. In *Proc. of the 4th KRDB Workshop, Athens, Greece*, 1998. http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-8/.

[9] T. Catarci and M. Lenzerini. Representing and using interschema knowledge in cooperative information systems. *Int. Journal of Intelligent and Cooperative Information Systems*, 2(4):375–398, 1993.

[10] W.W. Cohen, A. Borgida, and H. Hirsh. Computing least common subsumers in description logics. In *Proceedings AAAI-92*, pages 754–760. AAAI Press/The MIT Press, 1992.

[11] A.G. Cohn, L. Schubert, and S. Shapiro, editors. *Proceedings of the Sixth International Conference on Principles of Knowledge Representation and Reasoning (KR'98), Trento, Italy, June 2-5, 1998*, June 1998.

[12] F.M. Donini, M. Lenzerini, D. Nardi, and A. Schaerf. Reasoning in description logics. In G. Brewka, editor, *Principles of Knowledge Representation*. CSLI Publications, 1996.

[13] D. Fensel, M. Erdmann, and R. Studer. Ontobroker: The very high idea. In *Proceedings of the 11th International Flairs Conference (FLAIRS-98), Sanibal Island, Florida*, 1998.

[14] H. Garcia-Molina, Y. Papakonstantinou, D. Quass, A. Rajaraman, Y. Sagiv, J. Ullman, V. Vassalos, and J. Widom. The TSIMMIS approach to mediation: Data models and languages. *Journal of Intelligent Information Systems*, 1997.

[15] V. Haarslev, C. Lutz, and R. Möller. Foundations of spatioterminological reasoning with description logics. In Cohn et al. [11], pages 112–123.

[16] V. Haarslev, C. Lutz, and R. Möller. A description logic with concrete domains and a role-forming predicate operator. *Journal of Logic and Computation*, 9(3):351–384, June 1999.

[17] V. Haarslev and R. Möller. Expressive ABox reasoning with number restrictions, role hierachies, and transitively closed roles. In A.G. Cohn, F. Giunchiglia, and B. Selman, editors, *Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR'2000), Breckenridge, Colorado, USA, 2000*, April 2000.

[18] V. Haarslev, R. Möller, and A.-Y. Turhan. RACE User's guide and reference manual version 1.1. Technical Report FBI-HH-M-289/99, University of Hamburg, Computer Science Department, October 1999. Available at URL http://kogs-www.informatik.uni-hamburg.de/~haarslev/publications/report-FBI-289-99.ps.gz.

[19] V. Haarslev, R. Möller, and M. Wessel. The description logic $\mathcal{ALCNH}_{R+}$ extended with concrete domains. Technical Report FBI-HH-M-290/00, University of Hamburg, Computer Science Department, August 2000. Available at URL http://kogs-www.informatik.uni-hamburg.de/~haarslev/publications/report-FBI-290-00.ps.gz.

[20] I. Horrocks, U. Sattler, S. Tessaris, and S. Tobies. Query containment using a DLR ABox. LTCS-Report 99-15, LuFG Theoretical Computer Science, RWTH Aachen, Germany, 1999. See http://www-lti.informatik.rwth-aachen.de/Forschung/Reports.html.

[21] Ian Horrocks, Ulrike Sattler, and Stephan Tobies. Practical reasoning for expressive description logics. In Harald Ganzinger, David McAllester, and Andrei Voronkov, editors, *Proceedings of the 6th International Conference on Logic for Programming and Automated Reasoning (LPAR'99)*, number 1705 in Lecture Notes in Artificial Intelligence, pages 161–180. Springer-Verlag, September 1999.

[22] M. Klusch. *Cooperative Information Agents on the Internet (in German).* PhD thesis, University of Kiel, 1998.

[23] P. Lambrix, N. Shahmehri, and N. Wahlöf. A default extension to description logics for use in an intelligent search engine. In *Proc. of the 1st Hawaiian Int. Conf. on System Science*, pages 28–35, 1998.

[24] A.Y. Levy, A. Rajaraman, and J.J. Ordille. Query-answering algorithms for information agents. In *Proceedings, AAAI'96, 14th National Conference on Artificial Intelligence*, 1996.

[25] C. Lutz and R. Möller. Defined topological relations in description logics. In M.-C. Rousset et al., editor, *Proceedings of the International Workshop on Description Logics, DL'97, Sep. 27-29, 1997, Gif sur Yvette, France*, pages 15–19. Universite Paris-Sud, Paris, September 1997.

[26] Deborah L. McGuinness. *Frontiers in Artificial Intelligence and Applications*, chapter Ontological Issues for Knowledge-Enhanced Search. IOS-Press, Washington, DC, 1998.

[27] C. Meghini, F. Sebastiani, U. Straccia, and C. Thanos. A model of information retrieval based on a terminological logic. In *Proc. ACL SIGIR-93, Int. Conference on Research and Development in Information Retrieval, Pittsburg, PA*, pages 298–307, 1993.

[28] C. Meghini and U. Straccia. A relevance terminological logic for information retrieval. In *Proc. ACL SIGIR-96, Int. Conference on Research and Development in Information Retrieval, Zurich*, pages 197–205, 1996.

[29] R. Möller, V. Haarslev, and B. Neumann. Semantics-based Information Retrieval. In *Int. Conf. on Information Technology and Knowledge Systems*, Vienna, Budapest, August–September, 1998.

[30] R. Möller and M. Wessel. Terminological default reasoning about spatial information: A first step. In *Proc. of COSIT'99, International Conference on Spatial Information Theory, Stade*, pages 189–172. Springer Verlag, Berlin, 1999.

[31] D.A. Randell, Z. Cui, and A.G. Cohn. A spatial logic based on regions and connections. In B. Nebel, C. Rich, and W. Swartout, editors, *Principles of Knowledge Representation and Reasoning, Cambridge, Mass., Oct. 25-29, 1992*, pages 165–176. Morgan Kaufman, October 1992.

[32] J. Renz. A canonical model of the region connection calculus. In Cohn et al. [11], pages 330–341.

[33] M. Schmidt-Schauss and G. Smolka. Attributive concept descriptions with complements. *Artificial Intelligence*, 48(1):1–26, 1991.

[34] A.-Y. Turhan. Design and implementation of description logic provers for $\mathcal{ALC}(\mathcal{D})$ and $\mathcal{ALCRP}(\mathcal{D})$ (in German), October 1998. Bachelor Thesis (Studienarbeit).

[35] W.A. Woods and J.G. Schmolze. The KL-ONE family. In F. Lehmann, editor, *Semantic Networks in Artificial Intelligence*, pages 133–177. Pergamon Press, Oxford, 1992.